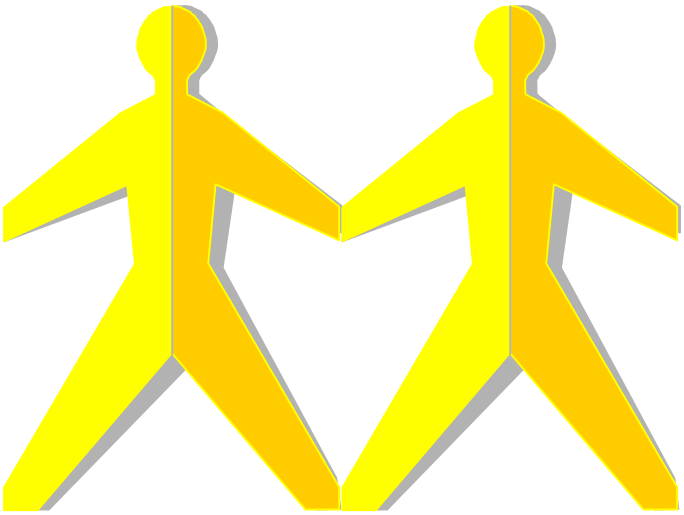


rvs®

Version 2.04.00.01

User Manual

The products listed in this manual are protected by copyright.

rvs®

Version 2.04.00.01

User Manual

© 2000 by gedas

Pascalstraße 11

10587 Berlin

This manual is protected by copyright. All rights reserved. No part of this book may be used or reproduced in any form or by any means including photocopies, microfilm or any other means or stored in a database or retrieval system without obtaining prior permission from gedas. Rights are also reserved as far as lectures, radio and television is concerned.

We reserve the right to make changes to the content of this manual without giving prior notice. gedas is not liable for technical or printing errors or defects in this manual. Moreover, gedas shall not be liable for damage which is directly or indirectly caused by delivery, performance and use of this material.

Contents

Contents 3
1 Introduction 6
1.1 Short Description of the System 6
1.2 Target Group 9
1.3 Structure of this Manual 9
1.4 Representation means 10
2 Overview rvs® 12
2.1 Data Set Names 13
2.2 Code Conversion 16
2.3 Serialization 16
2.4 Resident Receive Entries 17
2.5 Jobstart after Send Attempt 20
3 How to Work Interactively with rvs® 23
3.1 Input and Output Fields 23
3.2 Function Keys 25
3.3 Start Interactive Applications 26
3.3.1 Dialog Interface (`rvsdia`) for UNIX and AS/400 systems 26
3.3.2 rvsNT Adminstrator (`rvsmgr.exe`) for Windows NT 30
3.3.3 rvsNT (`rvsdiant.exe`) for Windows NT 31
3.4 Interactive Sending and Receiving for UNIX and AS/400 35
3.4.1 Create Send Requests 35
3.4.2 Display Send Requests and Received Transmissions 41
3.4.3 Operate Send Requests and Received Transmissions 57
3.5 Interactive Administration for UNIX and AS/400 60
3.5.1 Resident Receive Entries 60
3.5.2 Job Start after Send Attempt 77
3.5.3 User List 88
3.5.4 Information about rvs® 95
3.6 Interactive Sending and Receiving for Windows NT 96
3.6.1 Create Send Requests 96
3.6.2 Display Send Requests and Received Transmission 99
3.6.3 Operate Send Requests and Received Transmission 100
3.6.4 Select display criteria 103
3.7 Interactive Administration for Windows NT 105
3.7.1 Resident Receive Entries 105
3.7.2 Job Start after Send Attempt 110
3.7.3 User Entries 114
4 How to work with rvs® Batch Interface and rvs® C-CAL Interface (rvscal()) 116
4.1 Start the rvs® Batch Interface 116
4.2 Use the C-CAL Interface 118
4.2.1 Header File `rvscal.h` 118
4.2.2 Description of Commands 123
4.2.3 Syntax of Command Strings 124
4.2.4 Command `Start` 125
4.2.5 Command `End` 126
4.2.6 Command `Send` 126
4.2.7 Command `RESENTR` 130
4.2.8 Command `SENDJOB` 132
4.2.9 Command `USER` 134
4.2.10 Command `ACTIVATE` 135
4.2.11 Command `MODST` 135

© gedas deutschland GmbH / Competence Center EDI/rvs®

4.2.12 Command DELST 136
4.2.13 Command LISTPARM 136
4.2.14 Command SETPARM 137
5 How to Work with rvs® C-CAL Interface 138
5.1 Sending and Receiving with C-CAL Interface 138
5.1.1 Type Definitions 139
5.1.2 Get next send entry from Database 140
5.1.3 Get a send entry from Database 141
5.1.4 Set debug mode 141
5.1.5 Change status of SE 142
5.1.6 Get next information entry 143
5.1.7 Send a file 144
5.1.8 Create a Send Entry 145
5.2 Administration with C-CAL Interface 146
5.2.1 Functions to manage Station Table Entries 146
5.2.1.1 Type Definitions 146
5.2.1.2 Get next station entry from Database 148
5.2.1.3 Update station entry from Database 149
5.2.1.4 Get station entries from Database 149
5.2.1.5 Delete station entry from Database 149
5.2.1.6 Free all suspended Commands 150
5.2.1.7 Return Codes 150
5.2.2 Functions to manage rvs® Parameters 151
5.2.2.1 Type Definition 151
5.2.2.2 Get parameter value from Database 151
5.2.2.3 Get next parameter from Database 151
5.2.2.4 Writes parameter value into Database 152
5.2.2.5 Return Codes 152
5.2.3 Functions to manage rvs® Operator Commands 153
5.2.3.1 Store operator command into Database 153
5.2.3.2 Wake the Monitor 153
5.2.3.3 Return Codes 153
5.2.4 Functions to manage Resident Receive Entries 154
5.2.4.1 Type Definition and Macros 154
5.2.4.2 Get next command number of Resident Receive Entry from Database 154
5.2.4.3 Get Resident Receive Entry from Database 155
5.2.4.4 Configure Resident Receive Entries 155
5.2.4.5 Return Codes 156
5.2.5 Functions to manage Entries for Jobstart after Send Attempt 157
5.2.5.1 Type Definition and Macros 157
5.2.5.2 Get next command number of Job Start Entry from Database 157
5.2.5.3 Get Jobstart Entry from Database 158
5.2.5.4 Configure Entries for Jobstart after Send Attempt 158
5.2.5.5 Return Codes 159
5.2.6 Functions to manage User Entries 160
5.2.6.1 Type Definition and Macros 160
5.2.6.2 Get next User from Database 160
5.2.6.3 Get User Entry from Database 160
5.2.6.4 Configure User Entries 161
5.2.6.5 Return Codes 161
5.2.7 rvs® Database Functions 162
5.2.7.1 Type Definition and Macros 162
5.2.7.2 Dump Database 162
5.2.7.3 Recover Database 163
5.2.7.4 Initialize Database 163
5.2.7.5 Delete Database 164

5.2.7.6 Dump User-, Receive, Js- and Station Entries 164
5.2.7.7 Return Codes 165
5.2.7.8 Get the version of rvs® Database 165
5.2.7.9 Return Codes 165
5.2.8 Other Functions 166
5.2.8.1 Get SID from ODETTE ID or vice versa 166
5.2.8.2 List status of rvs® Commands 167
Glossary 168
Index 172

© gedas deutschland GmbH / Competence Center EDI/rvs®

1 Introduction

This chapter contains a short description of the rvs® system, an overview of the target group and the structure of the rvs® manuals as well as the explaination of the representation means which are used in this manual.

1.1 Short Description of the System

What rvs® is

The abbreviation rvs® stands for the German word Rechner-Verbund-System. The rvs® computer communication system is a well established base service for electronic data interchange, EDI.

Task of the system is, to guarantee transmisssion of electronic data between heterogenious computer platforms using different network protocols.

To do so, rvs® realizes a universal network model which can be configured by you within each network node.

It provides an efficient and reliable transport service for both standardized EDI message types and files of any format or content. You can get only data files which have been provided explicitly by rvs®. This means that rvs® does not provide an unauthorized access to remote data files.

The system was originally developed by Volkswagen AG and has been used in the German and European automobile industries for a number of years. rvs® uses the OFTP protocol. The Volkswagen AG has been developed an extansion to the standard OFTP: SNA LU6.2.

rvs® has become one of the most successful protagonists of EDI.

This "portable" version of rvs® has been developed in order to complement the classical rvs® product line which is based on MVS and VSE mainframes for use on midrange, mini systems and personal computing. Though the design and make of this product line differs considerably from the mainframe versions, the functional spectrum is almost exactly the same as that of rvs® MVS.

This version of rvs® is written in the language C and has been developed under the principles of modern design and the use of CASE-Tools.

What rvs® is not

rvs® is not an online system. It neither supports direct terminal-like access to other sites, nor does it provide a communication pipe from application to application on a data record level to the end user. You cannot directly execute transfers in your own application, you rather can place send orders in the rvs® database which will be handled asynchronously.

rvs® is not a job scheduling system.

rvs® does not care about the contents of the data sets it is transporting. It only acts as a transparent transport medium and performs no semantical interpretation of the data it carries.

rvs® is not a EDI converter. But additional components for converting between specific message formats (e.g. VDA, ODETTE, EDIFACT) can be purchased via gedas deutschland GmbH.

rvs® is not a network control or monitoring tool.

Supported Platforms

The following versions of portable rvs® are currently available:

- rvsX for SINIX using LU 6.2, X.25, ISDN and TCP/IP
- rvsX for AIX using LU 6.2, X.25, ISDN and TCP/IP
- rvsX for HP 9000 using LU 6.2, X.25, ISDN and TCP/IP
- rvsX for IRIX using ISDN and TCP/IP
- rvsX for Linux using ISDN and TCP/IP
- rvsX for SCO using ISDN and TCP/IP
- rvsX for Solaris using ISDN and TCP/IP
- rvs400 for IBM AS/400 systems using LU 6.2, X.25, ISDN and TCP/IP
- rvsNT for MS Windows NT using LU 6.2, X.25, ISDN and TCP/IP

© gedas deutschland GmbH / Competence Center EDI/rvs®

Basic Functional Characteristics

The main function of rvs® is the reliable transfer of files. rvs® is suitable both for the transfer of large files and for the transfer of many small files. rvs® supports many networks with many stations which are based on different platforms. Various data formats are supported thereby.

The key characteristics of rvs® are:

- rvs® works asynchronously, i.e. its processing is not under your direct control.
  You just places a send order in the rvs® database by means of a menu controlled user interface or out of a batch file or program. rvs® processes the send orders as soon as possible. The advantage is, that you or your application program does not need to wait for completion of a file transfer.
- The connection to the communication partner is automatically established when necessary.
- The automatic submission of jobs after reception of a data set is supported by means of resident receive entries. Wildcarding of data set names, user IDs, and station IDs is supported.
- The file transfer is possible for the following record formats:
  U Unstructed,
  T Text file,
  F Fixed,
  V Variable.
- Extensive security and authorization checking is performed.
- rvs® automatically repeats the transmission after an unsuccessful connection attempt or disruption of a connection. In the event of disruption rvs® only transmits the parts of a file which have not yet been transmitted. Therefore it is suited particularly to transmit large quantities of data even under difficult conditions.
- The compression of data during transfer increases transmission line throughput.
- Besides a menu controlled interface a single command line interface is provided. That means that you can call rvs® within batch or command files.
- A program C-CAL Interface allows you to call rvs® services out of a user application program.
- Message LOG files are provided for revision purposes.
- rvs® supports multiple languages in messages, operator (console) and user interfaces (presently English and German).

- The data conversion, if wanted, can be carried out. Text files are automatically converted to the code (EBCDIC or ASCII) used on the target partner system.
- The earliest date/time of a transmission can be defined for each file to be transferred.
- A serialization facility allows the transfer of files in a sequence which you can define.

1.2 Target Group

This manual is intended as a reference for persons who are assigned to work with rvs®.

The following skills are required to be able to use rvs®:

- knowledge of the current operating system
- understanding of the basic functions of rvs®
- knowledge of the current rvs® configuration as set up by the rvs® operator

Before starting to work it is advisable to have read this book.

1.3 Structure of this Manual

rvs® consists of the following manuals:

- Installation Manual
  The Installation Manual describes the installation of the rvs® software on different platforms. The **new installation** procedure and the **update installation** procedure will be explained step by step.
- User Manual
  The User Manual describes how to transfer files to partners and vice versa. The three (basic) interface methods – interactive, batch and API – are described. The available manual is theUser Manual.
- Operator Manual
  The Operator Manual describes how you can work with rvs® using the console. It contains tips how to maintain your data and a description of the utilities as well as how to configure rvs® after the installation. Furthermore, this manual contains information on the technical background of rvs®.
- Messages und Return Codes Manual
  This manual describes all messages and error codes which will be displayed on the rvs® Monitor and in the log file.

© gedas deutschland GmbH / Competence Center EDI/rvs®

The Installation Manual, the User Manual and the Operator Manual are available as printed and as electronic documents. The electronic document is available in Portable Document Format (PDF). The "Messages and Return Codes Manual" is only available as electronic document.

We have planned to create tutorials for the operator and the user. These manuals will contain all important steps how to work with rvs®. An explanation will be given step by step and in the correct order. That will help you to optimize your work with rvs®.

1.4 Representation means

This chapter contains the description of the indications which are used in this manual and the explanation of the expressions which are marked.

Indications

`courier`	commands, menu commands, file names, path names, programs, examples, scripts, qualifiers, data sets, fields, options, modes, window names, dialog boxes and statuses
BOLD and IN CAPITAL LETTERS	parameters, environment variables, variables
"quotation mark"	links to other manuals, sections and chapters, literature
bold	important, names of operating systems, proper names, buttons, function keys

Expressions

rvsX is the synonym of rvs® for **UNIX** systems.

rvsNT is the synonym of rvs® for **Windows NT** systems.

rvs400 is the synonym of rvs® for **AS/400** systems.

Directories

As user directories are found on different locations for the different operating systems we use the variable **$RVSPATH** in this manual. Default values are:

- `/home/rvs/` for **AIX**, **Solaris**, **IRIX**, **Linux** and **SCO**
- `/users/rvs/` for **HP-UX**
- `/defpath/rvs/` for **SINIX**
- `c:\rvs` for **Windows NT**

Substitute the variable with your correct path.

Generally, the file names on **OS/400** systems are always written in capital letters.

© gedas deutschland GmbH / Competence Center EDI/rvs®

2 Overview rvs®

rvs® is a reliable file transfer service, which can be used as a stand-alone utility to send, receive, and distribute data sets; rvs® can also be integrated into electronic document interchange (EDI) applications to automate data exchange between network nodes and users.

In accordance with different degrees of automation, there are three interfaces to rvs® to generate send and receive entries:

Dialog Interface is an interactive, full-screen utility to create individual entries one at a time; query functions tell you about the status of your requests.

Batch Interface reads commands from a data set. This input data set is a simple text file that can be prepared with any editor; or it might be generated as an output file by another application program.

C-CAL Interface allows application programs to generate rvs® command entries directly by calling C language functions.

Send entries are analyzed by rvs®. It creates send commands and starts sender processes which transfer your data set to the appropriate neighboring node(s) in the OFTP network for delivery or further routing. The send entry remains pending until positive acknowledgment has been received from the recipient.

Incoming files are accepted by the receiver processes and placed in temporary files. After that they are delivered to the to local recipient or send entries are created respectively to forward them to remote stations, if necessary.

Delivery of incoming files can be influenced by resident receive entries which tell rvs, under which name the file should be stored and whether a batch job should be started to further process incoming data.

With any of the above mentioned interfaces, one send entry will be created for each data set. Specifying entire subdirectories or libraries is not (yet) supported.

Please contact your local rvs® administrator to arrange, where you can find files that have been received and for which there is an active resident receive entry necessary.

2.1 Data Set Names

In this manual, we use the terms `file` and `data set` as synonyms.

The syntax for valid data set names is operating system specific. "Equivalent" names may be specified as

`/myid/invoice.dat`	under UNIX
`c:\myid\invoice.dat`	under OS/2 on a PC and under Windows NT
`MYID/DATA(INVOICE)`	under OS/400 on an AS/400
`MYID.INVOICE.DATA`	under MVS on an IBM host
`sysdrv:[userdisk.myid]invoice.data`	under VMS on a VAX
`. . .`	

These names are case sensitive on some systems (e.g. UNIX) and may or may not be case sensitive on others (e.g. OS/2, if the High Performance File System (HPFS) is being used). Security systems (such as RACF under MVS) may impose additional constraints on what is considered a legal data set name on a particular local system.

These differences may cause problems or inconveniences as described below when a file is sent to a different operation system.

© gedas deutschland GmbH / Competence Center EDI/rvs®

Virtual Data Set Name

For transfer and delivery, a file is identified by its Virtual Data Set Name (**VDSN**)[1]. You can specify this **VDSN** in the **NEW DSNAME** field when you interactively create a send request (see chapter "Create Send Requests" or in the **DSNNEW** parameter of the `SEND /CREATE` command (see "Command"). When you do not specify a **VDSN**, rvs® uses the name of the file being sent to generate one.

When a transmitted file is being delivered to its recipient, rvs® uses the **VDSN** as one criterion to look for matching resident receive entries. If none is found or the (best) matching entry does not define a data set name, **VDSN** is also used to generate a local data set name under which the file will be stored:

rvsMVS — uses **VDSN** as-is; so, when sending to an MVS host, make sure to specify a **VDSN** that adheres to MVS naming conventions and that starts with a high level qualifier that RACF is happy with. Do not use quotes (**"**) or apostrophes (**'**) to delimit your data set name.

portable rvs — uses **VDSN** to generate a name for the individual data set; the name of the path or library where this file will be stored is taken from the local rvs® environment (talk to your local rvs® administrator or see "Operator Manual" for more information on rvs® configuration).

VDSN is also one criterion to specify a job that should execute after a send attempt (see chapter "Job Start after Send Attempt" for more informations).

[1] VDSNs are used in the ODETTE file transfer protocol to pass the file name to the next node. Maximum length
(26 characters) and character set used when generating a default VDSN are due to the ODETTE protocol.

Time Stamping

When time stamping is requested, rvs® generates unique data set names when it delivers the file by using a numeric qualifier or by adding or replacing the last part of the name by a numerical value. For data sets with otherwise identical names, this number indicates the delivery sequence of the data sets (unless one or more old ones have been deleted, rvs® uses the smallest available number). The file systems currently supported by portable rvs® do not allow addition of a real time stamping, i.e. date and time of delivery, to the file name (with the exeption of UNIX systems and Windows NT).

Time stamping may be requested in a resident receive entry. It is done, when the data set is delivered.

© gedas deutschland GmbH / Competence Center EDI/rvs®

2.2 Code Conversion

You are probably aware that text files are stored on most systems in one of two computer codes, namely ASCII (American National Standard Code for Information Interchange) or EBCDIC (Extended Binary Coded Decimal Interchange Code).

While the assignment of digits and letters of the Latin alphabet is standardized within each of these two code families, special characters (like square brackets **[ ]**) or national language characters (like accented letters or umlauts) may be assigned different codes in different code pages within a family.

rvs® distinguishes between text files (format **T**) and non-text files (formats **F**, **V**, and **U**). Text files are always transmitted in ASCII and delivered in the local code of the target system. Non-text files are transmitted "as is" without any conversion, unless you specifically request code conversion by specifying **INPUT CODE** and **OUTPUT CODE** when creating the send entry. You also may involve a code conversion for received files by configuring a resident receive entry.

2.3 Serialization

Sometimes, it may be important, that a number of data sets arrive in a specific order at their destination to make sure they will be processed in proper sequence.

Even though rvs® will start processing send entries in the order in which they were submitted, there is no guarantee that the data sets will arrive in just this order (the line may break down while one file is being sent and another file may be transmitted before restart of the interrupted transmission is attempted).

To enforce the proper sequence,

- specify a **LABEL** when creating the send entry for the first data set in your series. This label consists of up to twenty characters of your choice.
- use the same label in the **LABEL** field and specify **SERIALIZE**=**Y** for all subsequent data sets.
- make sure, you create all send entries that are linked by a common label in the proper sequence.

rvs® will not start a serialized transfer until it has received the receipt for the predecessor. This will ensure transfer in the specified order.

2.4 Resident Receive Entries

A resident receive entry supersedes the default actions performed by rvs® when a data set is received from another rvs® station. It is characterized by the following parameters; all of them may have pattern[2] values.

DSN virtual data set name used for transmission

SID station ID of sender

UID user ID of sender; must be empty, if sending rvs® does not support distribution lists (in particular, this is true for MVS, VSE, and /36 hosts)

LUID user ID of local user to whom data set is being sent; must be empty, if sending rvs® does not support distribution lists. You may only specify a value different from your own user ID, if you are defined as a privileged user.

If a data set which matches all these four parameters is received by rvs®, the resident receive entry will be used by the rvs® monitor to determine how to deliver the received data set. There are additional parameters of a resident receive entry, which may be set at creation time:

ACCOUNT id used for local accounting

COMMENT a short text describing the purpose of the resident receive entry

DISP determine what should be done with the data set, when processing is completed

- **K** keep (and catalog) data set
- **D** delete data set
  Note, that the data set will be deleted, **BEFORE** the (optional) batch processing is completed. Therefore, specify **D** only, if you want to ignore the incoming file in any case.

DSNNEW a new local data set name that shall be used for the received data set

[2] A pattern contains wild card characters **?** or ***** matching one or indefinitely many characters, respectively.

© gedas deutschland GmbH / Competence Center EDI/rvs®

JOB name of a batch job data set which will be started, when the data set has been delivered[3].

This batch file may contain substitution patterns. rvs® substitutes them before submitting the job to the operating system for execution:

- ?**DSN**?: name of local data set, where received information has been stored
- ?**VDSN**?: virtual data set name under which the data set was transmitted
- ?**DTAVAIL**?: date, when the data set was available for sending
- ?**FORMAT**?: record format of received data set:
  - **F** fixed record length
  - **V** variable record length
  - **T** text file (character stream)
  - **U** unstructured file (byte stream)
- ?**MAXRECL**?: The meaning of this field depends upon the record format of the received data set:
  - **F** format: length of each record
  - **V** format: maximum length a record may have
  - **T** and **U** format: always **0** (zero)
- ?**BYTES**?: number of transmitted bytes
- ?**RECORDS**?: number of transmitted records for **F** and **V** format data sets; always zero for **T** and **U** format data sets
- ?**DTRCV**?: date, when data set was delivered to local user
- ?**LUID**?: recipient's (i.e. local) user ID
- ?**UID**?: sender's user ID
- ?**SID**?: sender's station ID
- ?**DSNTEMP**?: name of temp. data set (this data set should be deleted at the end of the job: "**DELETE** ?**DSNTEMP**?")

[3] The type of job submitted by rvs® depends on the operating system. E.g. under OS/2 a CMD data set will be executed as background process, whereas under OS/400 rvs® will start a job using **SBMDBJOB**.

REPLACE if **DISP**=**K** (keep), **REPLACE** determines the actions performed by rvs® when it has received a data set, that matches the name of an existing local data set

R replace existing data set

N create new data set name (this is the default)

I ignore incoming data set

TSTAMP make name of incoming data set unique by using a time stamp or other means

A resident receive entry can be created using the rvs® utilities via

- the dialog interface (chapter "How to Work Interactively with rvs®") by filling the corresponding mask,
- the batch interface (chapter "How to work with rvs® Batch Interface and rvs® C-CAL Interface (rvscal())"), or
- the C-CAL Interface (chapter "How to Work with rvs® C-CAL Interface").

© gedas deutschland GmbH / Competence Center EDI/rvs®

2.5 Jobstart after Send Attempt

A jobstart after send attempt entry supersedes the default actions performed by rvs® when ether a data set is sended to another rvs® station or a number of send attempts is made in vain. It is characterized by the following parameters; all of them may have pattern[4] values.

VDSN virtual data set name used for transmission;

The first two characters have the value **QS** in case of End-to-End-Responses (**EERP**) to distinguish between file transmission and **EERP** transmission. The value **QS*** i.e. determines the reaction if the transmission of an **EERP** failes.

SID station ID of receiver

An additional parameter controls the behaviour of this jobstart after send attempt entry.

ATTEMPTS determines in which case the job should start

- 0 starts in case of sucsessfull transmission (default)
- n starts if the specified number of send attempts fails

If a data set is tried to send by rvs® which matches these three parameters, rvs® monitor will start a job specified in following parameter:

[4] A pattern contains wild card characters **?** or ***** matching one or indefinitely many characters, respectively.

JOB name of a batch job data set which will be started[5].

This batch file may contain substitution patterns. rvs® substitutes them before submitting the job to the operating system for execution:

- ?**DSN**?: name of local data set, that has been sent; In case of an **EERP** we don't have a local data set name. The value of ?**DSN**? has the appearance **QS** (**SIDORIG** - **SIDDEST**) with meaning:
  SIDORIG sender's station ID
  SIDDEST receiver's station ID
- ?**VDSN**?: virtual data set name under which the data set was transmitted
- ?**DTAVAIL**?: date, when the data set was available for sending
- ?**FORMAT**?: Record format of the file sent
  - **F** fixed
  - **V** variable
  - **T** text
  - **U** unstructured
- ?**BYTES**?: Number of bytes transmitted
- ?**RECORDS**?: Number of records transmitted with **F** or **V** format
- ?**DTRCV**?: date, when data set was delivered to local user
- ?**LABEL**?: string if the send command contained a **LABEL** parameter. Can be used to identify the send command.

[5] The type of job submitted by rvs® depends on the operating system. E.g. under OS/2 a CMD data set and under Windows NT a batch file will be executed as background process, whereas under OS/400 rvs® will start a job using **SBMDBJOB**.

© gedas deutschland GmbH / Competence Center EDI/rvs®

- ?**SECN**?: command number of send command (**CN** of **SE**). Can be used to identify the send command
- ?**SKCN**?: Number of the send command
- ?**UID**?: sender's user ID; In case of **EERP** the value is always “!-**QS**-!”
- ?**SID**?: receiver's station ID
- ?**DSNTEMP**?: name of temp. data set (this data set should be deleted at the end of the job: "**DELETE** ?**DSNTEMP**?")

There are additional parameters of a jobstart after send attempt entry, which may be set at creation time:

COMMENT a short text describing the purpose of the resident receive entry;

A jobstart after send attempt entry can be created using the rvs® utilities via

- the batch interface (chapter "How to work with rvs® Batch Interface and rvs® C-CAL Interface (rvscal())"), or
- the C-CAL Interface (chapter "How to Work with rvs® C-CAL Interface")
- the rvsdia user interactive interface (chapter "How to Work Interactively with rvs®")

3 How to Work Interactively with rvs®

This screen interactive interface to rvs® lets you

- define send entries to transmit one data set at a time to one or more recipients at your local station or at one or more remote rvs® nodes.
- display the status of your transmissions.
- delete send entries, as long as rvs® did not start processing them, yet.
- create resident receive entries to influence the way incoming data is being stored or further processed.
- create resident send entries in order to trigger actions when files are sent or couldn't be sent respectively.
- modify or delete resident receive and send entries.

In the following sections, the masks (panels, screens, . .) are shown as they appear under the different operating system. Minor differences may exist for the different systems (e.g. **<ESC>** is not used on an **AS/400** screen). Each mask is followed by a help text describing the purpose of the screen and the meaning of it's fields. These help texts are also available as online help (see function key **<F1>**, below).

3.1 Input and Output Fields

When you are asked to provide information to rvs®, the panels will provide input fields which are separated by arrows (**===>**) from their descriptions; positions and maximum lengths of input fields are indicated by a series of underscores (_____) in the following sections.

Read only information is presented to you in output fields which are separated by colons (**:**) from their descriptions; positions and maximum lengths of output fields are indicated by periods (**.....**).

© gedas deutschland GmbH / Competence Center EDI/rvs®

Example

Mask SNDL on page 39 includes five input and two output fields:

Data set to be sent is		a **50** character output field; it is introduced with a colon. rvs® will display the name of the data set that you want to send in the part of the mask that is now filled with periods. Since you are specifying additional receivers, the data set name does not change and this field is read-only.
1.	**STATION ID**	describes a character input field, where the station ID of the node at which this new intended receiver is located can be entered.
2.	**USER**	
. . .		
3.	**TIMESTAMP**	are all input fields, also. If your entry is shorter then the maximum field size, you need not fill the field with blanks; trailing blanks are ignored.
.....		(the line below the list of supported function keys) is another output field as indicated by the periods. This is the message field where various information about successful or unsuccessful completion of system actions will be displayed. Warnings regarding missing or erroneous input values will appear here, too.

3.2 Function Keys

On most systems, you can use the following set of function keys to control the rvs® dialog interface. **Function key 1** will be called **<F1>** in this text, etc.

<F1> requests on-line help.

A brief explanation of the input field, where the cursor currently is located, will be displayed. To view the entire help text for the panel,

use function key **<F1>** to display context sensitive help, then press **<F1>** again; press **<ENTER>** to get to the next page; press **<ESC>** to leave help.

<F2> displays a mask to create a new entry (list displays).

<F3> exits panel, ignoring any input you may have made.

On systems that support an **Escape** key, **<ESC>** is equivalent to **<F3>**.

<F4> displays selection lists, if available.

In general, this key will work for fields, where you are asked to specifiy a station ID or a local data set name.

<F5> updates the current panel (when on-going transmissions are displayed) or confirmes that you really want to perform an action (like deleting a send request).

<F7> scrolls up (list displays).

If available, **<PageUp>** is equivalent to **<F7>**.

<F8> scrolls down (list displays).

If available, **<PageDown>** is equivalent to **<F8>**.

Depending on the system and terminal type, not all function keys required could be mapped to keys indicated as **F..** keys on the keyboard. If you have problems with function keys, you can use the numerical keypad fields on your keyboard, instead.

© gedas deutschland GmbH / Competence Center EDI/rvs®

3.3 Start Interactive Applications

This chapter describes the start of the interactive applications for the different operating systems.

For UNIX systems you can use the dialog interface to admnistrate and to receive and/or send files with rvs®. In additional, you have the Operator Console (see "Operator Manual") for the rvs® administration.

3.3.1 Dialog Interface (`rvsdia`) for UNIX and AS/400 systems

The dialog interface can be invoked as

`rvsdia [/e<envdsn>] [/l<language>] [/t]` for UNIX systems

where items within square brackets (**[...]**) are optional. The dialog interface can be used for adminstration and for sending and receiving tasks.

For AS/400 systems you start the dialog interface using the `rvs400` main window (command `WRKRVS`).

The command line parameters have the following meaning:

- **/e**: set alternate environment data set (for use by rvs® administrator only).
- **/l**: language: use message language given by character language
- **/t**: use test mode (for use by rvs® administrator only).

The dialog interface of UNIX systems starts out with the following initial mask (the masks for AS/400 are similar to those of UNIX):

RechnerVerbundSystem (mask INI)

```
 ------------------------------------------------------------------------
| ---------------------  Rechner Verbund System  -------------------     |
|                                                                        |
|                   RRRRRRR      VV     VV     SSSSSS                   |
|                   RR     RR    VV     VV    SS     S                  |
|                   RR     RR    VV     VV     SSS                      |
|                   RRRRRR       VV     VV      SSSS                    |
|                   RR  RRR       VV  VV         SSS                   |
|                   RR   RR        VVVV      S     SS                   |
|                   RR    RR         VV        SSSSSS                   |
|                                                                        |
|   OPTION ===>                                                          |
|                                                                        |
|         1 - send a dataset                                             |
|         2 - display / delete transmissions                             |
|         3 - delete send-requests                                       |
|         4 - resident receive entries                                   |
|         5 - send-job entries                                           |
|         6 - user entries                                               |
|         I - informations                                               |
|                                                                        |
| F1 = HELP, F3/ESC = EXIT                                               |
```

Select the action you want to perform next.

In most cases, follow-up panels will be displayed to ask for additional input or to display requested information. Active function keys are displayed in the bottom line and help you navigate the panels.

If you need help, press functionkey **< F1 >** to get more information about each field in the displayed panel.

SELECT OPTION

Select the number of the option that describes the task you want to perform:

`1 - send a data set:` A panel will be displayed, where you can specify

- name of data set to be sent,
- station ID and user ID of recipient,
- special processing options.

© gedas deutschland GmbH / Competence Center EDI/rvs®

`2 - display / delete transmissions:`

You will be able to specify selection criteria for the display of transmissions (both sent and received).

Select the transmission you are interested in

- to view details, (such as status, number of bytes, etc.)
- cancel a send request

`3 - delete send-requests:`

Choose this option if you know the parameters of the send request you want to delete; you will *not* see a selection list.

`4 - resident receive entries:`

You will be able to specify selection criteria for the display of resident receive entries you want to work with.

Select the resident receive entry (**RE**) you are interested in

- to view details,
- to modify any field in the **RE**, or
- to delete the entire **RE**.

Through this selection you will be able to create a new entry too.

`5 - job start after send attempt entries:`	You will be able to specify selection criteria for the display of job start after send attempt entries you want to work with. Select the job start after send attempt entry (**JS**) you are interested in • to view details, • to modify any field in the **JS**, or • to delete the entire **JS**. Through this selection you will be able to create a new entry too.
`6 - user entries:`	You will get a list of all rvs® users and you will be able: • to view details, • to modify a user entry, • to remove a user entry, • to create a new entry.
`I - Informations:`	This option provides access to additional on-line information about rvs®.

© gedas deutschland GmbH / Competence Center EDI/rvs®

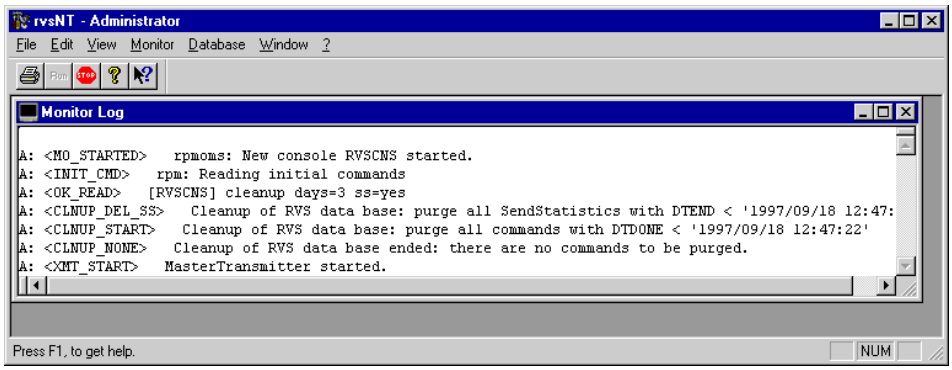
3.3.2 rvsNT Adminstrator (`rvsmgr.exe`) for Windows NT

The rvsNT administrator is used for configuring, administering and operating rvsNT. In order to carry out these activities the rvsNT Administrator is equipped with several detail windows. The configuration assumes administrator user rights at the system level and in the rvsNT environment.

Start the program `rvsNT` Administrator by invocation in the `rvs` file or by means of

`Start` → `Programs` → `rvs` → `rvs Administrator`

The rvsNT Administrator then opens with the `Monitor Log` window which displays the current rvs® Monitor activities.

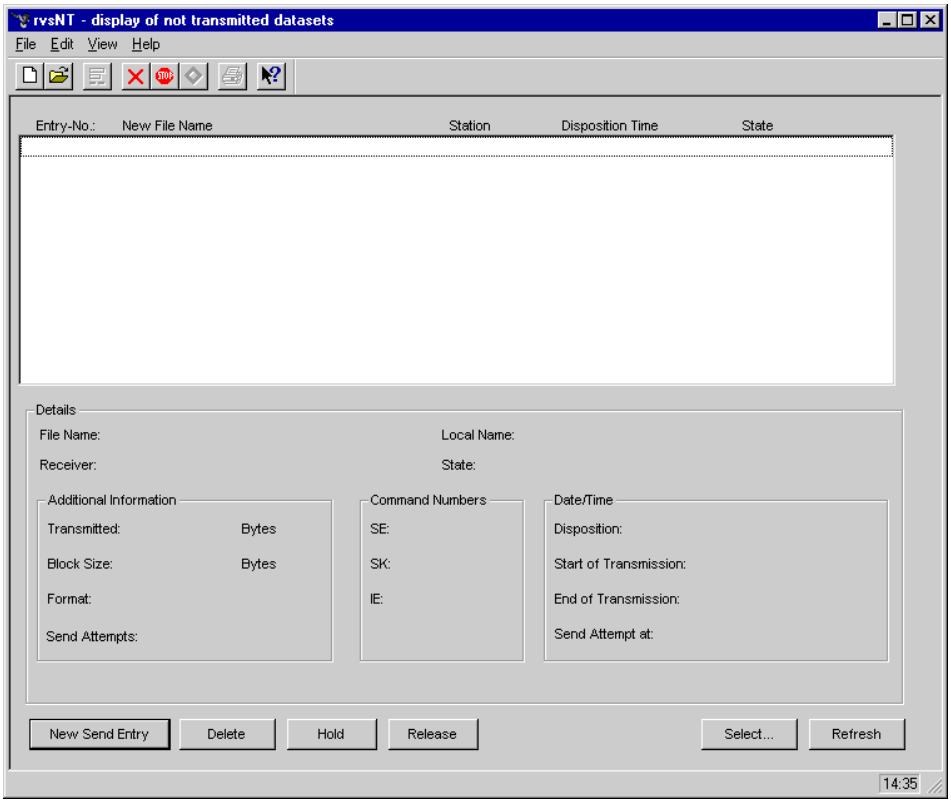


3.3.3 rvsNT (`rvsdiant.exe`) for Windows NT

Prerequisites: You should have a system which has been set up for operating rvsNT and the required access rights.

- Start the `rvsNT` program in the `rvs` file.

The `rvsNT` window then opens with display of not transmitted data sets.



The window displays the transmission information in four different display modes.

Display modes

- Send
- Receive
- Completely sent
- Completely received.

© gedas deutschland GmbH / Competence Center EDI/rvs®

Window areas

- Depending on the rvsNT display mode, the **entry list** will display the list of files which have been sent or received.
- The **details area** provides you with more detailed information on the currently selected file: file name, sender or recipient, local name or original name and status of the file.
- In the **additional information** you can see the currently transmitted data quantity, block size, file format and the number of dialling attempts (only for sending).
- The respective successive **command numbers** for send entries (SE), send command (SK) and information entries (IE) identify the individual transmission steps. These numbers can also be found in the rvsNT Log Book (file `c:\rvs\temp\rlco.log`).
- The rvs® Monitor logs the transmission time.
- Use the push buttons under the menu line to quickly execute the most frequently used commands. The push buttons change in accordance with the display mode.

Status display

The status field displays the progress of the send order selected or that of the order arriving.

Send

In the send or completely sent display modes the status field shows three events (SE SK QE); see table below.

Receive

In the receive or completely received display modes the status field shows three events (IE IZ QS); see table below.

Sample send order: `Status (qq-)`

`SE = q; SK = q; QE = *-*`

Abbr.	Event status	Possible values = status chain
	Send order	
SE	Send entry	q, f, p, e
SK	Send command	-, q, f, a, i, p, e
QE	Acknowledgement receipt, sent by recipient -, e	
	Reception arrival	
IE	Information receipt	q, f, p, e
IZ	Information assignment	q, f, p, e
QS	Acknowledgement transmission back to sender	-, q, f, a, i, p, e

Each event normally runs through the status chain in the order of possible values indicated. Some statuses have such a short duration that they are only briefly visible.

© gedas deutschland GmbH / Competence Center EDI/rvs®

The meaning of the values

- `-` no event
- `q` queued, not yet processed by Monitor
- `f` forwardable, waiting for transmission process
- `p` pending, waiting for completion
- `e` ended
- `a` active
- `i` in transit, transmitter currently processing order

In addition, every event may also have these statuses:

- `h` held by the system or operator
- `s` traffic with the destination suspended at present

Exit rvsNT

You can exit rvsNT at any time, even if all of the send orders have not been completed or you have not yet saved several of the files which have been received. The rvs® Monitor continues to process your send orders and places arriving files into your user directory. The next time you start rvsNT you can display the advance status of your send orders and any files which may have arrived.

- If you would like to exit rvsNT then use the `File` → `Exit` menu command.

3.4 Interactive Sending and Receiving for UNIX and AS/400

This chapter describes how to send and receive data sets with rvs® if you are using an ineractive application. First, it will be explained how you can create send requests. The next chapter describes how you can display sent and received transmissions and the last chapter shows how you can operate with send and receive data sets.

3.4.1 Create Send Requests

This chapter describes how you can create send requests and afterwards how you can transfer them.

Transmission of a Data set (mask SND)

```
------------------------------------------------------------------------------
|                R V S   -   TRANSMISSION OF A DATASET (SEND)                |
------------------------------------------------------------------------------
  Specify parameters below

  1. STATION ID  ===> |                 station id of receiver (blank = list)
  2. USER        ===>                      user id of receiver (blank=system)
  3. DSNAME      ===>
                                  name of dataset to be sent
  4. DISPOSITION ===> K           K = keep, D = delete
  5. FORMAT      ===>             T = text
  6. INPUT CODE  ===>             code of local dataset
  7. ACCOUNT     ===> bwa                account number of sender

  8. INIT TIME   ===>                      H=hold,expl.time (YYYY/MM/DD HH:MM)
  9. SERIALIZE   ===> N           serialize sending of datasets (Yes/No)
 10. LABEL       ===>                      used to serialize send requests

 11. NEW DSNAME  ===>                            rename
 12. OUTPUT CODE ===>             code of dataset at receiver
 13. TIMESTAMP   ===> N           add timestamp (Yes/No)

 F1 = HELP, F2 = ADDITIONAL RECEIVERS , F3/ESC = CANCEL , ENTER = PROCEED
```

Specify the necessary parameters to create a send entry for transmitting a data set.

1. STATION ID Specify station ID of receiver's rvs® node.

If you do not enter a value in this field, a list of available station IDs will be displayed.

Use key **< S >** to select a station ID from that list and confirm your choice by pressing **< ENTER >**.

Your choice will be inserted into the current panel.

© gedas deutschland GmbH / Competence Center EDI/rvs®

2. USER Specify user ID of receiver

An empty (blank) user ID indicates that the file should be delivered to the rvs® system at the target node, not to an individual user.

Transmissions to MVS, VSE, or /36 systems must not specify a recipient's user ID, because these systems do not support person-to-person transfers. This is also true, if your file must be routed through one of these systems.

3. DSNAME Specifies the name of the data set to be sent.

The data set must exist. This field must have an entry.

4. DISPOSITION Availability of data set after sending

Choose:

- **K** to keep data set after sending
- **D** to delete data set after sending

default: **K**

5. FORMAT Specifies type of data set and format of transmission

Choose:

- **T** = text file; a stream of ASCII characters
- **U** = unstructured (binary) byte stream
- **V** = variable record length
- **F** = fixed record length
- **blank** = defaults to systemformat (e.g. **U** for rvsNT and rvs2, **F** for rvs400)

6. INPUT CODE Specifies the code in which the data set is currently stored

Choose:

- **A** = ASCII
- **E** = EBCDIC
- **blank**:
  for **T** format data sets, native code of system is assumed (EBCDIC under OS/400; ASCII, otherwise) for other formats, code is considered unknown and no conversion will take place

7. ACCOUNT Specifies the account code of sender

Account code to which all activities relating to this send entry are to be charged.

default: account code of current session

8. INIT TIME Specifies the earliest time to send the data set

Choose:

- **H** = Data set is put in held status; it will not be sent until released by you or freed by the rvs® operator.
- **explicit time** = Year/Month/Day Hour:Minute (**YYYY**/**MM**/**DD HH**:**MM**)
- **blank** = now

9. SERIALIZE Specify, whether this data set belongs to a group of serialized transmissions (see **LABEL** field)

Choose:

- **Y**(es) for serialized sending of data set
- **N**(o) for non-serialized sending of data set

default: **N**

© gedas deutschland GmbH / Competence Center EDI/rvs®

10. LABEL

Name of group of serialized send requests.

User specified (descriptive) label for this entry. It is used to serialize on another send entry with the same label, if serialization was requested (when there is more than one other send entry with the same label, the latest one is used to serialize on).

11. NEW DSNAME

Rename data set for transmission.

This field specifies the name under which the file is known during transmission; resident receive entries at the destination must specify this name to further process the incoming data set.

If this field is left blank, the original data set name is used.

Note: Make sure to specify a valid MVS data set name when sending to an MVS-host!

12. OUTPUT CODE

Specify code in which data set should be delivered to recipient.

By default, T-format data sets are delivered in the 'native' code of the receiver's system; non-text files are delivered without any conversion, unless you specify 'A' or 'E' in this field.

Choose:

- **A** = ASCII
- **E** = EBCDIC
- **blank** = as explained in previous paragraph

default: **blank**

Lists of Stations (mask SIDL)

```
------------------------------------------------------------------------------
|                        R V S   -   LIST OF STATIONS                        |
------------------------------------------------------------------------------
  Option(s):  S - select

       Station-ID        Station name
------------------------------------------------------------------------------
  S    LOC               local rvs station
       R10               aaa (X.25)
       R11               VW AG (ISDN)
       R20               aaa (LU6.2)
       R30               aaa (TCP/IP)
       R40               aaa (X.25 via ISDN)
       RSX               remote station via X.25/ISDN
       TEST              remote station via X.25/ISDN
       ZZZ               echo test via X.25




------------------------------------------------------------------------------
     F1 = HELP, F3/ESC = CANCEL, RETURN = PROCEED
Row 1 .. 9 - End of table reached
```

This mask is displayed, whenever you request a selection list of rvs® nodes that can be reached from the local station (either directly or indirectly).

Functionkey **<F3>** : Leave display without selecting any station ID

Functionkey **<F7>** : Page up.

Functionkey **<F8>** : Page down.

Key **<ENTER>** : Confirm selection.

Specify option — If a station ID is selected by entering the line command **<S>**, the selected ID will be transferred into the appropriate field in the previous mask after you confirm your selection by pressing the **<ENTER>** key.

© gedas deutschland GmbH / Competence Center EDI/rvs®

Send data set: Distribution List (mask SNDDL)

The list of currently defined recipients of your data set is displayed.

Use line command option **<S>** to select a receiver for modification, use **<D>** to delete a receiver from the list.

Functionkey < F2 >: The propram displays a panel to define an additional receiver for this data set.

Functionkey < F3 > : Abort creation of the send entry; all information in the distribution list is discarded.

Functionkey < F7 > : Page up.

Functionkey < F8 > : Page down.

Key < ENTER > : Send data set.

Data set to be sent

Name of data set that should be delivered to one or more receivers.

Select option Enter an option in this column to modify or delete a receiver from the distribution list.

- Option S: Modify one or more parameters of this receiver.
- Option D: Delete this receiver from the distribution list.

STATION Station ID of receiver

USER User ID of of receiver

OUTPUT CODE Code in which data set will be sent.

- **A** = ASCII
- **E** = EBCDIC
- **blank** = default

TIME STAMP Data set name should be made unique by adding a time stamp.

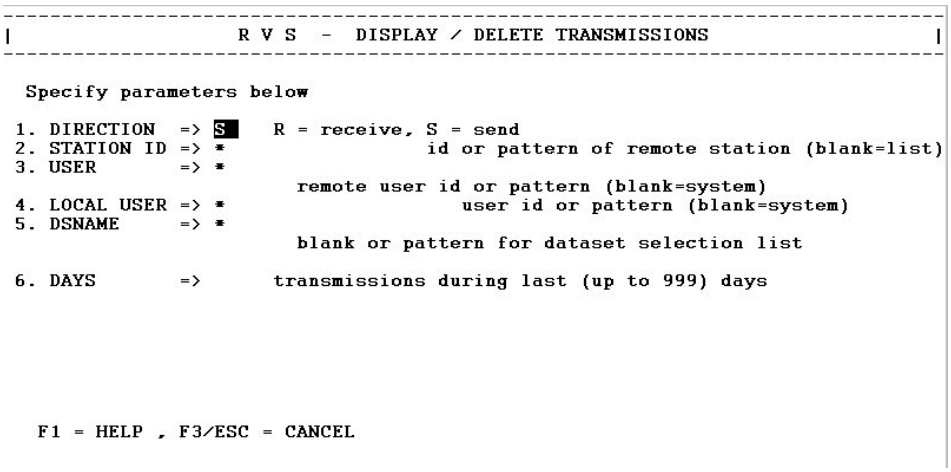
- **Y**(es) A time stamp will be added.
- **N**(o) No time stamp will be added.

NEW DSNAME Name in which the data set is to be delivered.

3.4.2 Display Send Requests and Received Transmissions

This chapter describes how you can display send requests and received transmissions.

Display / Delete Transmissions (mask DDT0)


R V S - DISPLAY / DELETE TRANSMISSIONS
Specify parameters below
1. DIRECTION => S R = receive, S = send
2. STATION ID => * id or pattern of remote station (blank=list)
3. USER => *
remote user id or pattern (blank=system)
4. LOCAL USER => * user id or pattern (blank=system)
5. DSNAME => *
blank or pattern for dataset selection list
6. DAYS => transmissions during last (up to 999) days
F1 = HELP , F3/ESC = CANCEL

Select one or more transmissions for display or deletion.

Functionkey < F3 > : Abort selection; return to previous panel.

Key < ENTER > : If station ID or data set name is blank, a selection list will be displayed. Otherwise, a list of matching entries will be shown.

1. DIRECTION Determines the direction of transmission.

Chooe:

- **R**(eceive) display information about received files.
- **S**(end) display send entries.

default: both

© gedas deutschland GmbH / Competence Center EDI/rvs®

2. STATION-ID Remote station ID.

Enter name or pattern of station ID of remote system.

If this field is blank, or **<F4>** is pressed while the cursor is positioned on this field, a list of available station IDs will be displayed. Use key **<S>** to select a station ID from that list and confirm your choise by pressing **<ENTER>**. Your choice will be inserted into the current panel.

3. USER User ID at remote system.

When you are looking at send entries, this is the receiver's name; for transmissions received at your station, this is the sender's user ID.

Enter name or pattern of user ID at remote system.

An empty user ID indicates that the file should be delivered to or was sent from the rvs® system at the target node, not to or from an individual user.

4. LOCAL USER User ID or pattern of local user.

When you are looking at send entries, this is the sender's name; for transmissions received at your station, this is the recipient's user ID.

5. DSNAME

Name or pattern of local data set name.

A pattern may contain wildcard characters asterisk (*****) and question mark (**?**). An asterisk matches any number of arbitrary characters, a question mark matches any one character.

If a pattern is specified, the list of transmissions will contain all matching entries.

If this field is blank, or **<F4>** is pressed while the cursor is positioned on this field, a list of available/matching data sets will be displayed. Use key **<S>** to select a data set and confirm your choise by pressing key **<ENTER>**.

This selected data set will be inserted in this field automatically.

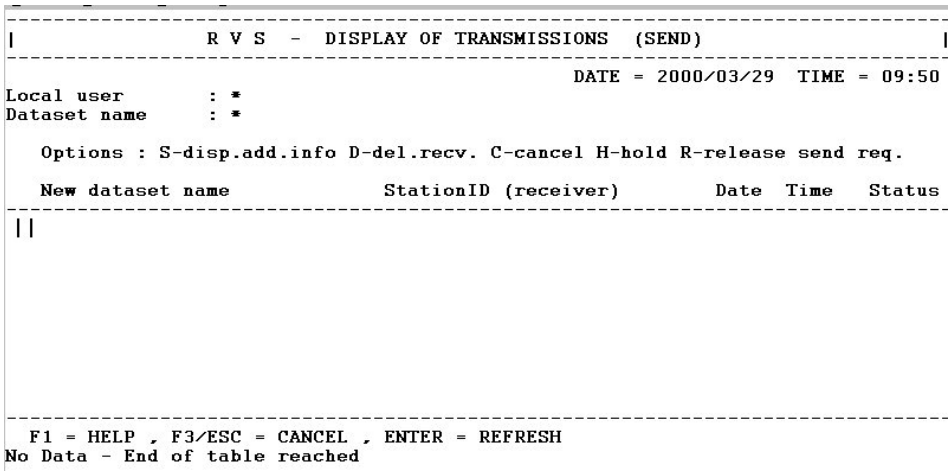
6. DAYS

Specify, how far back the system should look for matching entries.

You can specify at most **999** days; the period of time, however, for which information about completed transmissions is kept in the system is determined by frequency and retention period for database cleanups.

© gedas deutschland GmbH / Competence Center EDI/rvs®

Display of Transmissions (send) (mask DDT1)


| R V S - DISPLAY OF TRANSMISSIONS (SEND) |
DATE = 2000/03/29 TIME = 09:50
Local user : *
Dataset name : *
Options : S-disp.add.info D-del.recv. C-cancel H-hold R-release send req.
New dataset name StationID (receiver) Date Time Status
F1 = HELP , F3/ESC = CANCEL , ENTER = REFRESH
No Data - End of table reached

Information about send entries is displayed.

You may request additional information (line command option **<S>**) or you may modify those entries for which processing has not yet started.

Functionkey < F1 > : Displays help information.

Functionkey < F2 > : Add a new send entry.

Functionkey < F3 > : Terminate and return to previous panel.

Functionkey < F5 > : Refresh status of active transmissions.

Functionkey < F7 > : Page up.

Functionkey < F8 > : Page down.

DATE Current date

TIME Current time

Local user Name or pattern of sender's (local) user ID.

Data set name Name of tranfer data set.

Select option

- Option S: display additional information about this transmission.
- Option C: cancel send request
- Option H: put send request in hold status
- Option R: release send request for transmission

<D>, **<C>**, and **<H>** can only be used as long as processing of send entry has not yet started.

<R> can only be used if request is held (status **h**)

New data set name

Name under which data set is known during transmission (**VDSN** = Virtual Data Set Name).

For single recipient, this is the name, under which it will be delivered.

SID

Station ID of receiver.

Date

Earliest point in time that this transmission should be executed, as defined when this entry was created (**month**/**day**).

Time

Earliest point in time that this transmission should be executed, as defined when this entry was created (**hour**:**minute**).

© gedas deutschland GmbH / Competence Center EDI/rvs®

STAT

Status of transmission at this time.

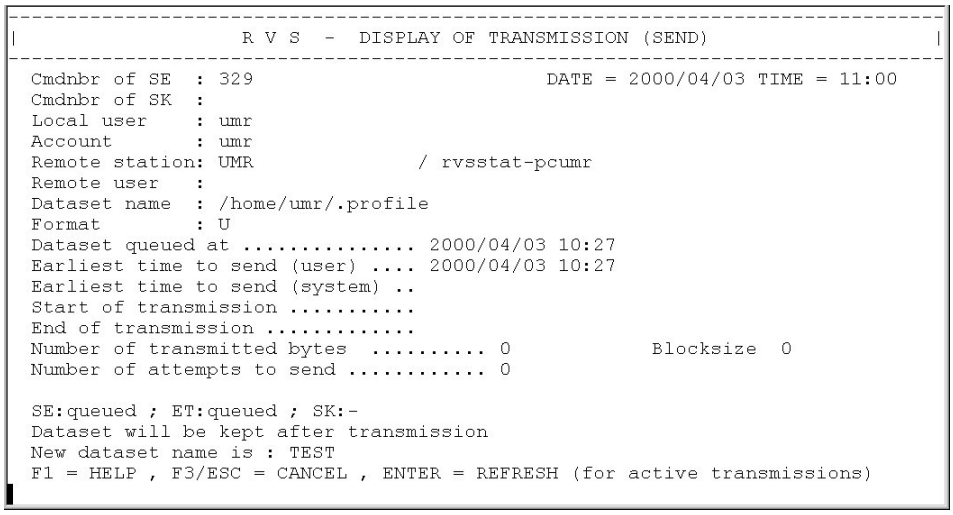
This field may show up to three statuses (e.g. **EEE** after processing is completed), namely status of:

- SE: command that controls processing of entire send entry; when it ends, the data set has reached its recipient.
- ET: description of one (local or remote) recipient (status can be **Q**, **E**, or **D**).
- SK: command that controls transmission to the neighboring node.

Commands may be in one of these statuses:

- **D** = deleted (by operator or user)
- **E** = ended (normal termination)
- **H** = set in hold status (by operator or user)
- **P** = pending (e.g. SK is waiting for receipt)
- **Q** = queued (waiting to be processed)
- **S** = suspended (all traffic to the neighboring node has been suspended by local rvs® operator) (SK, only)
- **I** = in transfer (ongoing transmission)
- **F** = forwardeable (transmission may start)

Display of Transmission (send) (mask DSPT)



```
|                  R V S  -  DISPLAY OF TRANSMISSION (SEND)                  |
Cmdnbr of SE  : 329                          DATE = 2000/04/03 TIME = 11:00
Cmdnbr of SK  :
Local user    : umr
Account       : umr
Remote station: UMR              / rvsstat-pcumr
Remote user   :
Dataset name  : /home/umr/.profile
Format        : U
Dataset queued at ............... 2000/04/03 10:27
Earliest time to send (user) .... 2000/04/03 10:27
Earliest time to send (system) ..
Start of transmission ...........
End of transmission .............
Number of transmitted bytes  ......... 0           Blocksize  0
Number of attempts to send ............ 0

SE:queued ; ET:queued ; SK:-
Dataset will be kept after transmission
New dataset name is : TEST
F1 = HELP , F3/ESC = CANCEL , ENTER = REFRESH (for active transmissions)
```

The purpose of this panel is, to display a specified send transmission. As long as this transmission is active, you can refresh the status of transmission by pressing functionkey **<F5>**.

Functionkey **<F3>** : The program returns to the previous panel.

Functionkey **<F5>** : Refresh the status of active transmissions.

Cmdnbr of SE

Internal number under which this send request is being processed.

You may need this number when communicating with your local rvs® operator (e.g. to have a send request deleted for which processing already started).

Cmdnbr of SK

Internal number of the send command (SK) created from the send request (SE). Evaluates to **0**, if no SK has yet been created.

You may need this number when communicating with your local rvs® operator (e.g. to have a send request deleted for which processing already started).

© gedas deutschland GmbH / Competence Center EDI/rvs®

DATE	Current date
TIME	Current time
Local user	Local user ID of sender.
Account	Account to which all rvs® activities relating to this send request are being charged.
Remote station (ID)	Station ID of remote station.
Remote station (name)	Name of remote station.
Remote user	Remote user ID.
Data set name:	Name of data set to be transmitted.
Format	Format in which data set is sent. • **T** = text file; a stream of ASCII characters • **U** = unstructured (binary) byte stream • **V** = variable record length • **F** = fixed record length • **blank** = Systemformat
Data set queued at (date)	Date of creation of this send entry (**year**/**month**/**day**)

Data set queued at (time)	Time of creation of this send entry (**hour**:**minute**)
Earliest time to send (user) (date)	Earliest point in time that this transmission should be executed as specified by the user when the entry was created. (**year**/**month**/**day**)
Earliest time to send (user) (time)	Earliest point in time that this transmission should be executed as specified by the user when the entry was created. (**hour**:**minute**)
Earliest time to send (system) (date)	Earliest point in time (system defined) that this transmission should be executed. (**year**/**month**/**day**) For 'queued' commands, this is the time when rvs® will attempt to execute the command; for 'pending' or 'ended' commands, this is the last time, when rvs® attempted to execute it.
Earliest time to send (system) (time)	Earliest point in time (system defined) that this transmission should be executed. (**hour**:**minute**)
Start of transmission (date)	Date when transmission started. (**year**/**month**/**day**)

© gedas deutschland GmbH / Competence Center EDI/rvs®

Start of transmission (time)	Time when transmission started. (**hour**:**minute**)
End of transmission (date)	Date when transmission ended. (**year**/**month**/**day**)
End of transmission (time)	Time when transmission ended. (**hour**:**minute**)
bytes / blocks	Units (bytes or blocks) in which amount of transmitted data is measured.
Number of transmitted bytes/blocks	Total number of bytes or blocks transmitted up to now. This value is updated, whenever **SENDBLOCKS** buffers or blocks have been transferred (see rvs® Operator Manual for a description of parameter **SENDBLOCKS**).
Blocksize	Blocksize of transmitted file (zero for file formats **T** and **U**).
Number of attempts to send:	Total number of attempts (both successful and unsuccessful) to send file to neighboring rvs® node.
status line	Status of transmission will be displayed in line 20.

disposition line Disposition of data set will be displayed in line 21.

new-dsn line Virtual data set name (i.e. the name under which the data set is known during transmission) will be displayed in line 22.

Display of Transmissions (receive) (mask DDTR)

```
--------------------------------------------------------------------------------
|                    R V S  -  DISPLAY OF TRANSMISSIONS (RECEIVE)               |
--------------------------------------------------------------------------------
                                                DATE = 2000/03/29  TIME = 09:51
Local user       : *
Dataset name     : *

Options : S-display additional information

   Dataset name (as received)   Station-ID (sender)         Date  Time   STATUS
--------------------------------------------------------------------------------
 ■




--------------------------------------------------------------------------------
  F1 = HELP, F3/ESC = CANCEL, ENTER = REFRESH
No Data - End of table reached
```

Information about receive entries is displayed.

Functionkey **<F3>** : Terminate and return to previous panel.

Functionkey **<F5>** : Refresh status of active receiver.

Functionkey **<F7>** : Page up.

Functionkey **<F8>** : Page down.

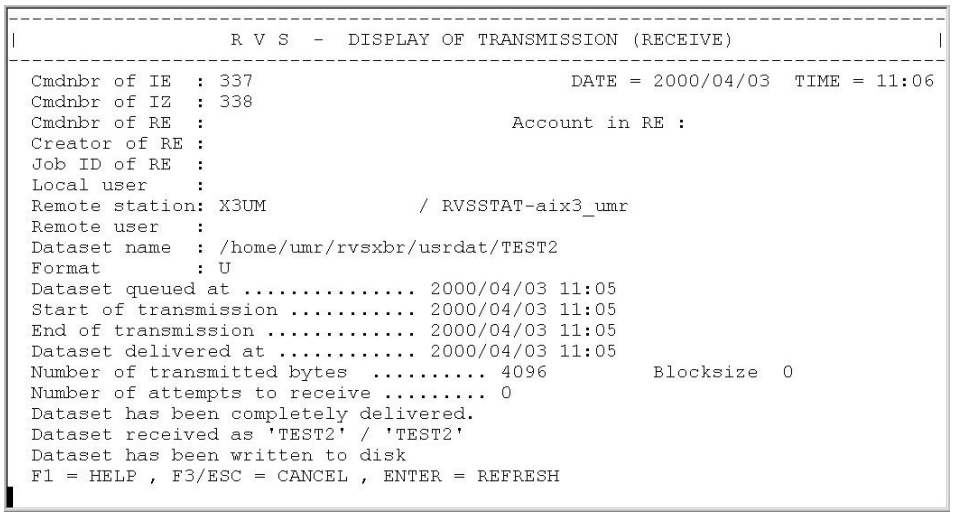
DATE Current date

TIME Current time

© gedas deutschland GmbH / Competence Center EDI/rvs®

Local user	Local user ID whose entries are shown.
Data set name	Name of local data set.
Select option	Option S: display additional information about this received data set.
Data set name (as received)	Name of data set as received (virtual data set name).
Station ID of sender	User ID of sender
Date	Date, when data set was delivered.
Time	Time, when data set was delivered.
STATUS	Status of received data set. This field may show up to three statuses (e.g. **EEE** after processing is completed): • `IE`: command that controls processing of entire transmission; when it ends, a receipt for the sender is created. • **ET**: description of one (local or remote) recipient (status can be **Q** or **E**). • `IZ`: command that controls delivery to one local recipient. Commands may be in one of these statuses: • **D** = deleted (by operator) • **E** = ended (normal termination) • **H** = set in hold status (by operator) • **P** = pending (e.g. `IE` may be waiting for all deliveries to complete) • **Q** = queued (waiting to be processed)

Display of Transmission (receive) (mask DSPR)



```
|                 R V S  -  DISPLAY OF TRANSMISSION (RECEIVE)                 |
--------------------------------------------------------------------------------
  Cmdnbr of IE  : 337                          DATE = 2000/04/03  TIME = 11:06
  Cmdnbr of IZ  : 338
  Cmdnbr of RE  :                          Account in RE :
  Creator of RE :
  Job ID of RE  :
  Local user    :
  Remote station: X3UM            / RVSSTAT-aix3_umr
  Remote user   :
  Dataset name  : /home/umr/rvsxbr/usrdat/TEST2
  Format        : U
  Dataset queued at ............... 2000/04/03 11:05
  Start of transmission ........... 2000/04/03 11:05
  End of transmission ............. 2000/04/03 11:05
  Dataset delivered at ............ 2000/04/03 11:05
  Number of transmitted bytes  .......... 4096         Blocksize  0
  Number of attempts to receive ......... 0
  Dataset has been completely delivered.
  Dataset received as 'TEST2' / 'TEST2'
  Dataset has been written to disk
  F1 = HELP , F3/ESC = CANCEL , ENTER = REFRESH
```

The purpose of this panel is, to display a specified received data set. As long as this transmission is active, you can refresh the status of transmission by pressing functionkey **<F5>**.

Functionkey **<F3>** : The program returns to the previous panel.

Functionkey **<F5>** : Refresh the status of active transmissions.

Cmdnbr of IE Internal number under which this data has been received in your local system.

DATE Current date

TIME Current time

Cmdnbr of IZ Internal number under which this data set has been delivered to a local user.

Cmdnbr of RE Internal number of resident receive entry, if this received data set has been processed by a matching resident receive entry

© gedas deutschland GmbH / Competence Center EDI/rvs®

Account of RE	Account to which all rvs® activities relating receiving and storing a data set is to be charged, if resident receive entry is specified for this received data set.
Creator of RE	User ID of creator of resident receive entry, if specified for this received data set.
Job ID of RE	ID of batch job that further processed received information, if a resident receive entry is specified for this data set.
Local user	Local user ID.
Remote station (ID)	Station ID of remote station.
Remote station (name)	Name of remote station.
Remote user	Remote user ID.
Data set name	Local name of data set that has been delivered.
Format	Format in which data set is received. • **T** = text file; a stream of ASCII characters • **U** = unstructured (binary) byte stream • **V** = variable record length • **F** = fixed record length • **blank** = Systemformat

Data set queued at (date)	Date of creation for sending of this data set on remote station. (**year**/**month**/**day**)
Data set queued at (time)	Time of creation for sending of this data set on remote station. (**hour**:**minute**)
Start of transmisson (date)	Date when transmission on remote station was started. (**year**/**month**/**day**)
Start of transmission (time)	Time when transmission on remote station was started. (**hour**:**minute**)
End of transmission (date)	Date of end of transmission on remote system. (**year**/**month**/**day**)
End of transmission (time)	Time of end of transmission on remote system. (**hour**:**minute**)
Data set delivered at (date)	Date when data set was delivered on local system. (**year**/**month**/**day**)
Data set delivered at (time)	Time when data set was delivered on local system. (**hour**:**minute**)

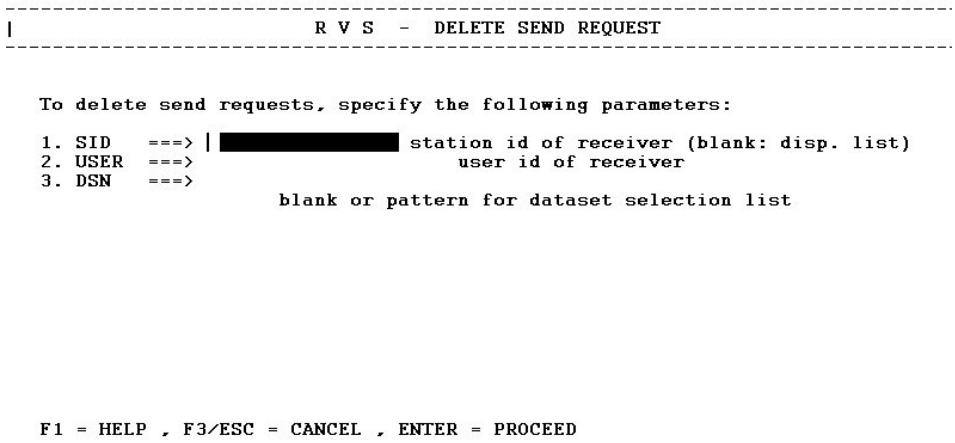
© gedas deutschland GmbH / Competence Center EDI/rvs®

bytes / blocks	Units (**bytes** or **blocks**) in which amount of transmitted data is measured.
Number of transmitted bytes/blocks	Total number of bytes or blocks transmitted up to now. This value is updated, whenever **RECVBLOCKS** buffers or blocks have been transferred (see rvs® operator manual for a description of parameter **RECVBLOCKS**).
Blocksize	Blocksize of transmitted file (zero for file formats **T** and **U**).
Number of attempts to receive	Total number of attempts (both successful and unsuccessful) to receive file from neighboring rvs® node.
status line	Status of transmission will be displayed in line 20.
Data set received as	Virtual data set name (i.e. the name under which the data set is known during transmission) will be displayed in line 21. There may be two different virtual data set names, if the sender sent the original data set to different users. In that case, the first **VDSN** is the label for the entire send request, and the second one is the **VDSN** used for delivery to you.
disposition line	Disposition of data set will be displayed in line 22, e.g. whether the data set has been ignored or written to disk.

3.4.3 Operate Send Requests and Received Transmissions

This chapter describes how you can operate data sets with rvs® if you are using an ineractive application. It will be explained how to delete send requests.

Delete Send Request (mask DELSR)


R V S - DELETE SEND REQUEST
To delete send requests, specify the following parameters:
1. SID ===>
station id of receiver (blank: disp. list)
2. USER ===>
user id of receiver
3. DSN ===>
blank or pattern for dataset selection list
F1 = HELP , F3/ESC = CANCEL , ENTER = PROCEED

In this panel, you specify a send entry that you want to delete. All fields must be specified.

Functionkey **<F3>** : Cancel delete request; no send entry will be deleted.

Key **<ENTER>** : When all fields have been specified, **<ENTER>** will check, whether such a send entry exists and can still be deleted (because it has not been processed yet). If the entry is available, a delete / confirm panel will be displayed.

If **SID** or **DSN** is still empty, a selection list will be shown.

1. SID Station ID of receiver

If this field is blank, a list of available station IDs will be displayed. Use key **<S>** to select a station ID from that list and confirm your choice by pressing **<ENTER>**. Your choice will be inserted into the current panel.

© gedas deutschland GmbH / Competence Center EDI/rvs®

2. USER User ID of receiver

3. DSN Name of data set

Enter name of data set or blank or pattern for data set selection list

Confirm Delete of Send Request (mask CDSE1)

You are asked to confirm that you really want to delete the selected send entry.

Functionkey **<F3>** : Cancel delete request; the send entry will be kept.

Functionkey **<F5>** : The displayed send request will be deleted and the file will not be transmitted.

Station ID of receiver Station ID of receiver.

User ID of receiver User ID of receiver.

Data set name Name of data set to be sent.

List of Data sets for Transmission (mask DSNL)

This panel lists all available (local) data sets for transmission. It is displayed whenever you request a data set selection list. Select a data set to include its name in the previous panel.

Functionkey **<F3>** : Leave this list without selecting a data set and return to the previous mask.

Functionkey **<F7>** : Page up.

Functionkey **<F8>** : Page down.

Key **<ENTER>** : Confirm selection.

Specify option	If a data set name is selected by entering the line command **<S>**, the selected name will be transferred into the appropriate field in the previous mask after you confirm your selection by pressing the **<ENTER>** key.
Local DSN	Name of the local data set.

© gedas deutschland GmbH / Competence Center EDI/rvs®

3.5 Interactive Administration for UNIX and AS/400

This chapter describes how to administrate resident receive entries, jobs after send attempts and user lists as well as how to call for information of rvs®.

You must call the application `rvsdia` if you want to work interactively with rvs®.

3.5.1 Resident Receive Entries

This capter describes how to operate resident receive entries, how to start a job after send attempt, how user lists can be administrated and how you can get informations about rvs®.

Resident Receive Entries (mask RE)

```
          R V S  _  DISPLAY / UPDATE / DELETE RESIDENT RECEIVE ENTRY

To display/change/delete a resident receive entry specify parameters:

1. SID         ===> *                station id of sender (name or pattern)
2. USER        ===> *                  user id of sender (name or pattern)
3. DSN         ===> *
                    name or pattern of dataset as received (blank: all)
4. LOCAL USER ===> *                   user id of local user
                    (name or pattern, privileged users only)




F1 = HELP , F3/ESC = CANCEL , ENTER = show list of entries
```

The purpose of this panel is to display, change, or delete an existing resident receive entry. Specify the following parameters to describe the resident receive entry you want to access.

Functionkey **<F3>** : Cancel request; all receive entries remain unchanged.

1. SID Station ID of sender

If this field is blank, a list of available station IDs will be displayed. Use key **<S>** to select a station ID from that list and confirm your choise by pressing **<ENTER>**. Your choice will be inserted into the current panel.

This is one of the selection criteria; incoming data set must have originated at a station with a matching name for this receive entry to be further considered as a possible action for the incoming file.

To be actually used, all selection criteria must match.

2. USER User ID of sender

Name or pattern of user ID of sender.

One of the selection criteria; incoming data set must have originated from a sender with a matching name for this receive entry to be further considered as a possible action for the incoming file.

To be actually used, all selection criteria must match.

3. DSN Name of received data set

Enter name of data set of resident receive entry. Leave blank or specify pattern to get a data set selection list.

One of the selection criteria; incoming data set must be known by a matching name for this receive entry to be further considered as a possible action for the incoming file. To be actually used, all selection criteria must match.

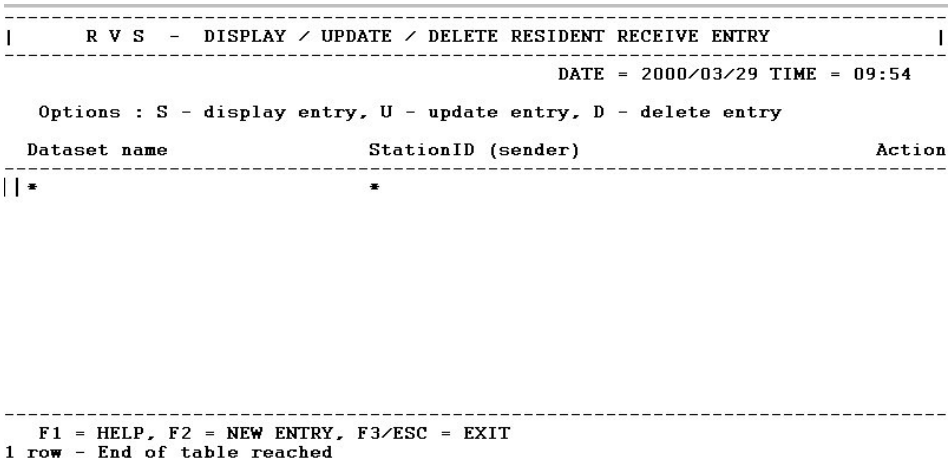
© gedas deutschland GmbH / Competence Center EDI/rvs®

4. LOCAL USER Name or pattern of local user ID who is the recipient of the incoming information.

You may specify something other than your own user ID, only if you are defined as privileged user.

Default: your current user ID

List of Resident Receive Entries (mask REL)


R V S - DISPLAY / UPDATE / DELETE RESIDENT RECEIVE ENTRY
DATE = 2000/03/29 TIME = 09:54
Options : S - display entry, U - update entry, D - delete entry
Dataset name
StationID (sender)
Action
*
*
F1 = HELP, F2 = NEW ENTRY, F3/ESC = EXIT
1 row - End of table reached

This panel displays a list of resident receive entries.

Use line command **<S>**, **<U>**, or **<D>** to get more information about the entry, update, or delete it, respectively.

Functionkey **<F1>** : Display help information.

Functionkey **<F2>** : Display a mask to add a new entry.

Functionkey **<F3>** : Ignore any selection and leave display.

Functionkey **<F7>** : Page up.

Functionkey **<F8>** : Page down.

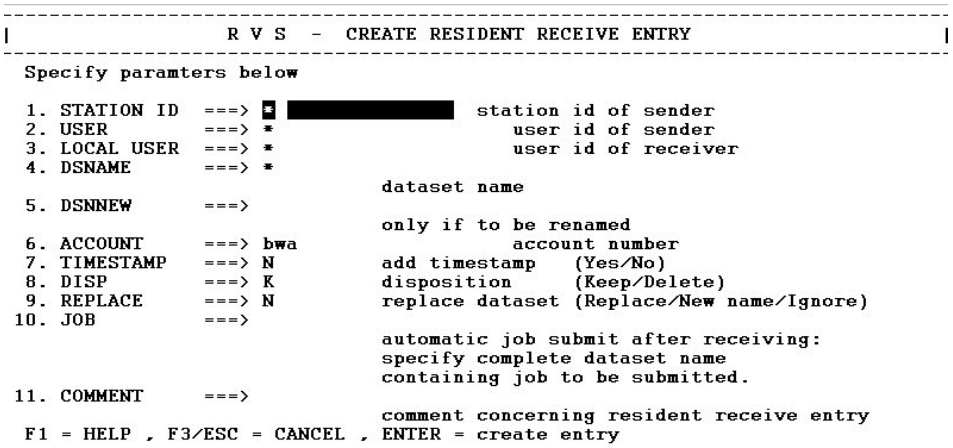
Key **<ENTER>** : Confirm selection.

DATE Current date

TIME	Current time
Specify option	Enter one of these options to the left of one of the displayed resident receive entries: • **S**: Display resident receive entry. • **U**: Update resident receive entry. • **D**: Delete resident receive entry.
DATA SET-NAME	Name or name pattern of incoming data set that will be processed by this resident receive entry.
SID	Station ID of sender.
UID (sender)	User ID of sender.
Local User	Name or pattern of local user ID who is the recipient of the incoming information.
Action	Indicates last action you performed against the list entry (**UPDATE**, **DELETE**) in the current session.

© gedas deutschland GmbH / Competence Center EDI/rvs®

Create Resident Receive Entry (mask CRRE)


R V S - CREATE RESIDENT RECEIVE ENTRY
Specify paramters below
1. STATION ID ===> * station id of sender
2. USER ===> * user id of sender
3. LOCAL USER ===> * user id of receiver
4. DSNAME ===> *
dataset name
5. DSNNEW ===>
only if to be renamed
6. ACCOUNT ===> bwa account number
7. TIMESTAMP ===> N add timestamp (Yes/No)
8. DISP ===> K disposition (Keep/Delete)
9. REPLACE ===> N replace dataset (Replace/New name/Ignore)
10. JOB ===>
automatic job submit after receiving:
specify complete dataset name
containing job to be submitted.
11. COMMENT ===>
comment concerning resident receive entry
F1 = HELP , F3/ESC = CANCEL , ENTER = create entry

In this panel, you can create a resident receive entry.

In a resident receive entry, a user may specify what actions should be taken, when a data set has been received.

When changes become necessary, a new entry will be created and the old one will be deleted.

Functionkey **<F3>** : No resident receive entry will be created.

Key **<ENTER>** : If all fields are specified the resident receive entry will be created.

1. STATION ID Station ID of sender

If this field is blank, a list of available station IDs will be displayed. Use key **<S>** to select a station ID from that list and confirm your choice by pressing **<ENTER>**. Your choice will be inserted into the current panel.

This is one of the selection criteria; incoming data set must have originated at a station with a matching name for this receive entry to be further considered as a possible action for the incoming file. To be actually used, all selection criteria must match.

2. USER

User ID of sender.

Name or pattern of user ID of sender.

One of the selection criteria; incoming data set must have originated from a sender with a matching name for this receive entry to be further considered as a possible action for the incoming file. To be actually used, all selection criteria must match.

3. LOCAL USER

Name or pattern of local user ID who is the recipient of the incoming information.

You may specify something other than your own user ID only, if you are defined as privileged user

Default: your current user ID

To match a transfer from an MVS host, this field must be left blank or it must contain an asterisk (*****) to match any ID, because rvsMVS sends to your station, not to an individual user.

One of the selection criteria; incoming data set must be intended for a local user with a matching name for this receive entry to be further considered as a possible action for the incoming file.

To be actually used, all selection criteria must match.

4. DSNAME

(Virtual) name of incoming data set.

One of the selection criteria; incoming data set must be known by a matching name for this receive entry to be further considered as a possible action for the incoming file.

To be actually used, all selection criteria must match.

© gedas deutschland GmbH / Competence Center EDI/rvs®

5. DSNNEW

Specify name under which data set should be stored.

This is the only way in which you as a user can specify directory or library where the incoming file is to be placed. Be careful, however, if you used a pattern in any of the selection criteria, above; **DSNNEW** cannot contain any wild cards and more than one matching data set may arrive before you processed the old one.

If you do not specify a fully qualified data set name, rvs® will try to generate one, using information from your current path or library.

The full name will be shown when the panel is redisplayed after the resident receive entry has been created. Please check that the name is what you intended it to be.

Make sure that (sub-)directories or libraries exist when files are being delivered. rvs® will not create directories or libraries and deliveries will fail, if they do not exist.

6. ACCOUNT

Account to which all rvs® activities relating to receiving and storing a data set is to be charged.

7. TIMESTAMP

Generate unique data set name when delivering file by adding a time stamp as dsn qualifier.

On systems where this is not possible, numerical values will be used as last qualifiers.

Choose :

- **Y** = yes
- **N** = no

Default: **N**

8. DISP

Determines what should be done with the data set when processing completes.

Choose :

- **K**(eep) Data set will be kept after processing (and cataloged, if these are seperate actions on the local system).
- **D**(elete) Data set will be deleted after processing. This option only takes effect if there are jobs to process.

Default: **K**

9. REPLACE

Specifies what should be done, when a data set with the same name already exists.

Choose :

- **R**(eplace) This option is only meaningful, if **DISPOSITION** = **K** and a data set with the specified name does already exist.
- **N**(ew) Create a new data set name.
- **I**(gnore) Incoming data set will be ignored.

Default: **N**

10. JOB

Name of data set that can be started as a batch job, to specify a following process. e.g.

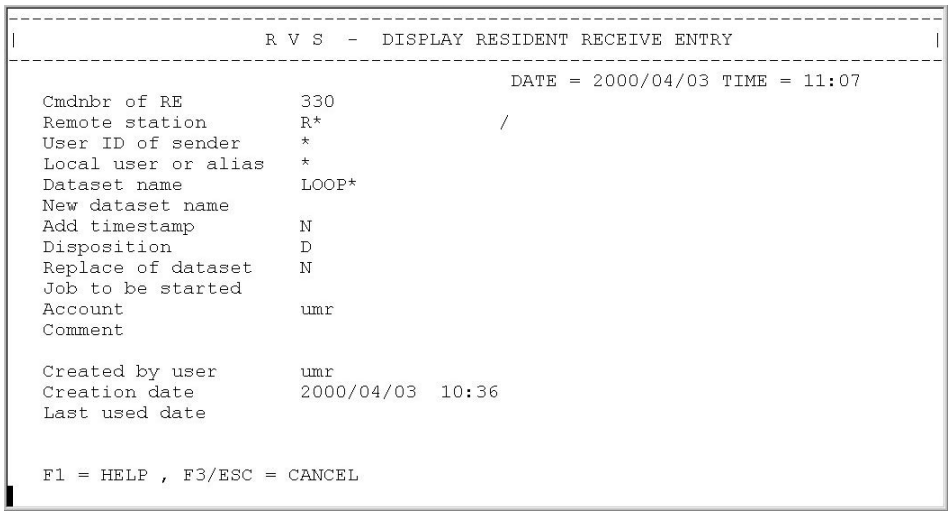
- ____**.bat** on MS-DOS
- ____**.CNTL** on MVS

11. COMMENT

Brief comment describing the purpose of this entry (used for display only).

© gedas deutschland GmbH / Competence Center EDI/rvs®

Display Resident Receive Entry (mask RE1)


R V S - DISPLAY RESIDENT RECEIVE ENTRY
DATE = 2000/04/03 TIME = 11:07
Cmdnbr of RE 330
Remote station R* /
User ID of sender *
Local user or alias *
Dataset name LOOP*
New dataset name
Add timestamp N
Disposition D
Replace of dataset N
Job to be started
Account umr
Comment
Created by user umr
Creation date 2000/04/03 10:36
Last used date
F1 = HELP , F3/ESC = CANCEL

This panel displays details about the resident receive entry you selected in the previous panel.

Functionkey **<F3>** : Leave display.

DATE Current date

TIME Current time

Cmdnbr of RE Internal number of resident receive entry.

Remote station Station ID or pattern of remote station.

One of the selection criteria; incoming data set must have originated at a station with a matching name for this receive entry to be further considered as a possible action for the incoming file.

To be actually used, all selection criteria must match.

Remote station (id) Name of remote station.

User ID of sender One of the selection criteria; incoming data set must have originated from a sender with a matching name for this receive entry to be further considered as a possible action for the incoming file.

To be actually used, all selection criteria must match.

Local user or alias User ID of local recipient.

One of the selection criteria; incoming data set must be intended for a local user with a matching name for this receive entry to be further considered as a possible action for the incoming file.

To be actually used, all selection criteria must match.

Data set name (Virtual) name under which the file is known during transfer.

One of the selection criteria; incoming data set must be known by a matching name for this receive entry to be further considered as a possible action for the incoming file.

To be actually used, all selection criteria must match.

New data set name Name under which the data set will be stored in the local system.

Add timestamp

- **Y**(es) Generate a unique data set name when delivering the file.
- **N**(o) Otherwise

© gedas deutschland GmbH / Competence Center EDI/rvs®

Disposition

- **K**(eep) Data set will be kept after processing (and cataloged, if these are seperate actions on the local system).
- **D**(elete) Data set will be deleted after processing (e.g. in combination with print data).

Replace of data set

- **R**(eplace) Replace existing data set
- **N**(o) Create new data set name
- **I**(gnore) Ignore incoming data set

Job to be started

Name of data set that will be started as a batch job when data is received.

Account

Account to which all rvs® activities relating to receiving and storing a data set is to be charged.

Comment

Brief comment describing the purpose of this entry (used for display, only).

Created by user

User ID of person who created this entry.

Creation date (date)

Date when this entry was created.

Creation date (time)

Time when this entry was created.

Last used date (date)

Date when this entry was last used to determine the actions that occured when data was received.

Last used date (time)	Time when this entry was last used to determine the actions that occured when data was received.

Confirm Delete Resident Receive Entry (mask CDRE)

In this mask, you are asked to confirm that you really want to delete this displayed resident receive entry.

Functionkey **<F3>** :	Cancel delete request; the displayed entry will remain active.
Functionkey **<F5>** :	The displayed entry will be deleted.

DATE	Current date
TIME	Current time
Remote station (id)	Station ID of the remote system One of the selection criteria; incoming data set must have originated at a station with a matching name for this receive entry to be further considered as a possible action for the incoming file. To be actually used, all selection criteria must match.
Remote station (name)	(Descriptive) name of the remote system.

© gedas deutschland GmbH / Competence Center EDI/rvs®

User ID of sender

One of the selection criteria; incoming data set must have originated from a sender with a matching name for this receive entry to be further considered as a possible action for the incoming file

To be actually used, all selection criteria must match.

Local user or alias

User ID of local recipient.

One of the selection criteria; incoming data set must be intended for a local user with a matching name for this receive entry to be further considered as a possible action for the incoming file.

To be actually used, all selection criteria must match.

Data set name

(Virtual) name under which the file is known during transfer.

One of the selection criteria; incoming data set must be known by a matching name for this receive entry to be further considered as a possible action for the incoming file.

To be actually used, all selection criteria must match.

Comment

Brief comment describing the purpose of this entry (used for display only).

Created by user

User ID of person who has created this entry.

Creation date (date)

Date when this resident receive entry was created.

Creation date (time)	Time when this resident receive entry was created.
Last used date (date)	Date when this entry was last used to determine the actions that occured when data was received.
Last used date (time)	Time when this entry was last used to determine the actions that occured when data was received.

Update Resident Receive Entry (mask REU)

The purpose of this panel is to update an existing resident receive entry. This mask is similar to the mask `Create Resident Receive Entry`, CRRE.

To update the displayed entry, overwrite text in those fields you want to change and confirm by pressing key **<ENTER>**. The old entry will be deleted and a new one will be created.

Functionkey **<F3>** :	Cancel modification request; the displayed resident receive entry will remain unchanged.
Key **<ENTER>** :	Update the existing resident receive entry.

© gedas deutschland GmbH / Competence Center EDI/rvs®

1. STATION ID Station ID of sender

To change this field, overwrite it with a new station ID; or blank it out to get a list of all availiable statio-ids to select a new one.

This is one of the selection criteria; incoming data set must have originated at a station with a matching name for this receive entry to be further considered as a possible action for the incoming file.

To be actually used, all selection criteria must match.

2. USER User ID of sender.

One of the selection criteria; incoming data set must have originated from a sender with a matching name for this receive entry to be further considered as a possible action for the incoming file.

To be actually used, all selection criteria must match.

3. LOCAL USER Name or pattern of local user ID who is the recipient of the incoming information.

You may specify something other than your own user ID, only if you are defined as privileged user.

Default: your current user ID

This is one of the selection criteria; incoming data set must be intended for a local user with a matching name for this receive entry to be further considered as a possible action for the incoming file.

To be actually used, all selection criteria must match.

4. DSNAME

Name of incoming data set

(Virtual) name under which the file is known during transfer.

One of the selection criteria; incoming data set must be known by a matching name for this receive entry to be further considered as a possible action for the incoming file.

To be actually used, all selection criteria must match.

5. DSNNEW

New name of data set that should be given to the recipient's copy of the received data.

6. ACCOUNT

Account in which all rvs® activities relating to receiving and storing a data set is to be charged.

7. TIMESTAMP

Data set name should be made unique by adding of time stamp.

Choose :

- **Y** = yes
- **N** = no

Default: **N**

8. DISP

Determines what should be done with the data set when processing completes.

Choose :

- **K**(eep) Data set will be kept after processing (and cataloged, if these are seperate actions on the local system).
- **D**(elete) Data set will be deleted after processing. This option only takes effect if there are jobs to process.

Default: **K**

© gedas deutschland GmbH / Competence Center EDI/rvs®

9. REPLACE

Specifies what should be done, when a data set with the same name already exists.

Choose :

- **R**(eplace) This option only takes effect, if **DISPOSITION** = **K** and a data set with the specified name does already exist.
- **N**(ew) Create a new data set name.
- **I**(gnore) Incoming data set will be ignored.

Default: **N**

10. JOB

Name of data set that can be started as a batch job, to specify a following process. e.g. ____**.bat** on MS DOS ____**.CNTL** on MVS

11. COMMENT

Brief comment describing the purpose of this entry (used for display only).

3.5.2 Job Start after Send Attempt

This chapter contains a description of how to operate with job starts after send attempt entries.

Job Start after Send Attempt Entries (mask JS)

```
|       R V S  -  LIST / UPDATE / DELETE OF SEND-JOB-ENTRIES                  |

 specify parameter of send-job-entries:


 1. STATION-ID        ===>                              station id of receiver
 2. USER-ID           ===>  *                           user id of sender
 3. DATASETNAME       ===>  *                           virtual datasetname
 4. SENDATTEMPTS      ===>  *                           number of sendattempts




 F1 = HELP, F3/ESC = CANCEL, EINGABE = show list of entries
```

The purpose of this panel is to select criteria, you want to access. Specify the following parameters to describe the job start after send attempt entry you want to access.

Functionkey **<F1>** : Display help information.

Functionkey **<F3>** : Cancel request; all receive entries remain unchanged.

1. SID Station ID of receiver

If this field is blank, a list of available station IDs will be displayed. Use key **<S>** to select a station ID from that list and confirm your choise by pressing **<ENTER>**.

Your choice will be inserted into the current panel.

© gedas deutschland GmbH / Competence Center EDI/rvs®

2. USER User ID of sender

Name or pattern of user ID of sender.

One of the selection criteria; outgoing data set must have originated from a user with a matching name for this JobStart entry to be further considered as a possible action for the outgoing file.

To be actually used, all selection criteria must match.

3. DSN Name of data set to send

Enter name of data set of JobStart after SendAttempt entry.

Leave blank or specify pattern to get a data set selection list.

4. ATTEMPTS Number of (failed) send attempts before starting a job.

List of Job Start after Send Attempt Entries (mask JSSL)

This panel displays a list of JobStart after SendAttempt Entries.

Use line command **<S>**, **<U>**, or **<D>** to get more information about the entry, update, or delete it, respectively.

Functionkey **<F1>** : Display help information.

Functionkey **<F2>** : Display a mask to add a new entry.

Functionkey **<F3>** : Ignore any selection and leave display.

Functionkey **<F7>** : Page up.

Functionkey **<F8>** : Page down.

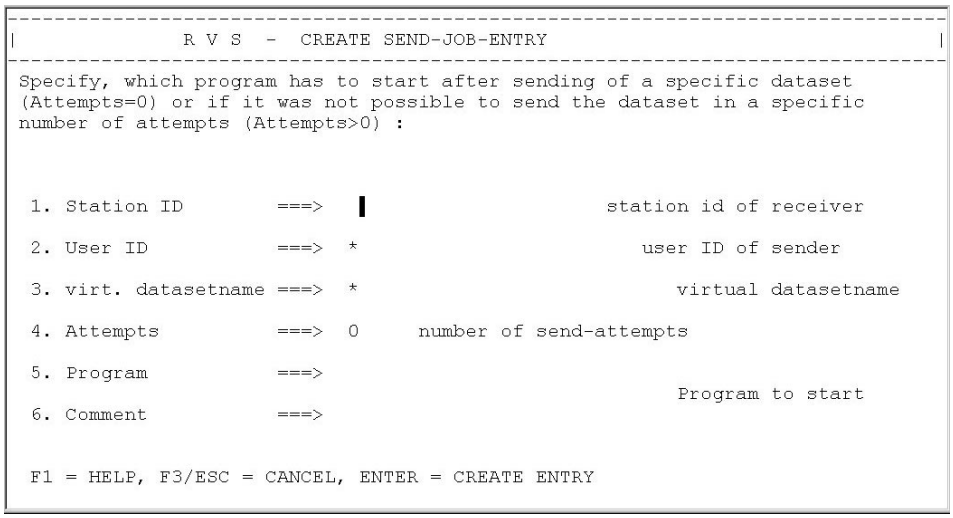
Key **<ENTER>** : Confirm selection.

DATE Current date

TIME	Current time
Specify option	Enter one of these options to the left of one of the displayed JobStart after SendAttempt entries: • **S**: Display the entry. • **U**: Update the entry. • **D**: Delete the entry.
SID	Station ID of receiver.
Data set name	Name or name pattern of outgoing data set that will be processed by this JobStart entry.
Job Name	Job that will be started after the specified amount of sendattempts.
Attempts	Number of Sendattempts before the specified job will be started.

© gedas deutschland GmbH / Competence Center EDI/rvs®

Create Job Start after Send Attempt Entry (mask CJSS)


R V S - CREATE SEND-JOB-ENTRY
Specify, which program has to start after sending of a specific dataset
(Attempts=0) or if it was not possible to send the dataset in a specific
number of attempts (Attempts>0) :
1. Station ID ===>
station id of receiver
2. User ID ===> *
user ID of sender
3. virt. datasetname ===> *
virtual datasetname
4. Attempts ===> 0
number of send-attempts
5. Program ===>
Program to start
6. Comment ===>
F1 = HELP, F3/ESC = CANCEL, ENTER = CREATE ENTRY

In this panel, you can create job start after send attempt entry.

In a job start after send attempt entry, a user may specify what actions should be taken, when a data set has been attempt to send.

Functionkey **<F1>** : Display help information.

Functionkey **<F3>** : No entry will be created.

Key **<ENTER>** : If all fields are specified the entry will be created.

1. STATION ID Station ID of receiver.

If this field is blank, a list of available station IDs will be displayed. Use key **<S>** to select a station ID from that list and confirm your choice by pressing **<ENTER>**.

Your choice will be inserted into the current panel.

This is one of the selection criteria; outgoing data set must have destinated to a station with a matching name for this entry to be further considered as a possible action for the outgoing file. To be actually used, all selection criteria must match.

2. USER

User ID of sender.

Name or pattern of user ID of sender.

One of the selection criteria; outgoing data set must have originated from a sender with a matching name for this entry to be further considered as a possible action for the outgoing file.

To be actually used, all selection criteria must match.

Attention: The user ID as selection criteria is not yet supported.

3. DSNAME

(Virtual) name of outgoing data set.

One of the selection criteria; outgoing data set must be known by a matching name for this entry to be further considered as a possible action for the outgoing file.

To be actually used, all selection criteria must match.

4. ATTEMPTS

One of the selection criteria; specify amount of send-attempts after which the specified job should be started.

If **ATTEMPTS** is greater than 0, the job will be started after the specified amount of failed send attempts.

If **ATTEMPTS** is 0, the job will be started after a successfull transmission.

Default: **0**

© gedas deutschland GmbH / Competence Center EDI/rvs®

5. JOB Name of data set that can be started as a batch job, to specify the following process.

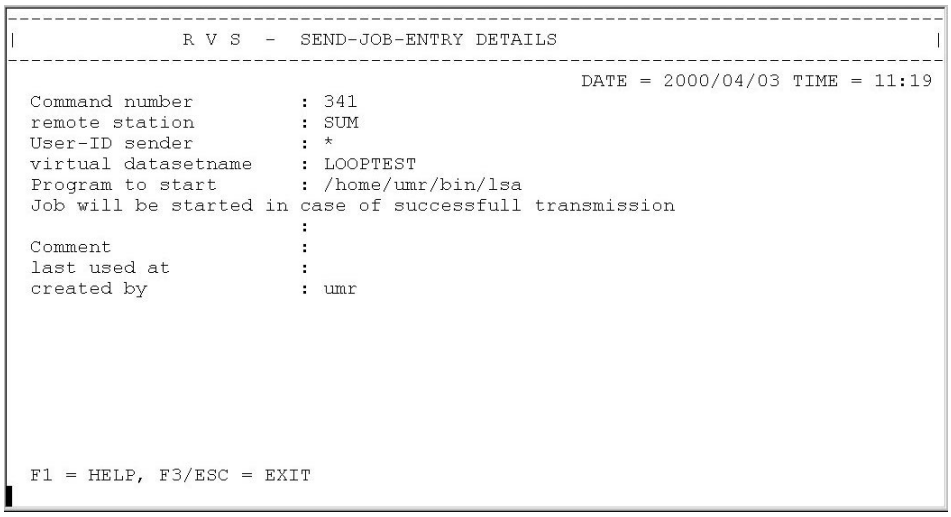
e.g.

- ____**.bat** on MS-DOS or WindowsNT
- ____**.CNTL** on MVS
- ____**.cmd** on OS2

On AS/400 the job will be started with the OS/400 system command: SBMDBJOB.

6. COMMENT Brief comment describing the purpose of this entry (used for display only).

Display Job Start after Send Attempt Entry (mask SJSS)


R V S - SEND-JOB-ENTRY DETAILS
DATE = 2000/04/03 TIME = 11:19
Command number : 341
remote station : SUM
User-ID sender : *
virtual datasetname : LOOPTEST
Program to start : /home/umr/bin/lsa
Job will be started in case of successfull transmission
:
Comment :
last used at :
created by : umr
F1 = HELP, F3/ESC = EXIT

In this panel, displays details about the JobStart after SendAttempt entry you selected in the previous panel.

Functionkey **<F1>** : Display help information.

Functionkey **<F3>** : Leave display and return to previous panel.

DATE	Current date
TIME	Current time
CmndNbr.	Internal number of this entry.
Station ID	Station ID of receiver. If this field is blank, a list of available station IDs will be displayed. Use key **<S>** to select a station ID from that list and confirm your choice by pressing **<ENTER>**. Your choice will be inserted into the current panel. This is one of the selection criteria; outgoing data set must have destinated to a station with a matching name for this entry to be further considered as a possible action for the outgoing file. To be actually used, all selection criteria must match.
User	User ID of sender. Name or pattern of user ID of sender. One of the selection criteria; outgoing data set must have originated from a sender with a matching name for this entry to be further considered as a possible action for the outgoing file. To be actually used, all selection criteria must match. Attention: The user ID as selection criteria is not yet supported.

© gedas deutschland GmbH / Competence Center EDI/rvs®

virtual DS-Name

(Virtual) name of outgoing data set.

One of the selection criteria; outgoing data set must be known by a matching name for this entry to be further considered as a possible action for the outgoing file.

To be actually used, all selection criteria must match.

Job to submit

Name of data set that can be started as a batch job, to specify the following process.

e.g.

- ____**.bat** on MS DOS or Windows NT
- ____**.CNTL** on MVS
- ____**.cmd** on OS2

On AS/400 the job will be started with the OS/400 system command: `SBMDBJOB`.

Job will be started after

One of the selection criteria; specifies amount of send-attempts after which the specified job should be started.

Comment

Brief comment describing the purpose of this entry (used for display only).

Last used at

Date and time when this entry was last used to detremine the actions that occured when a data set was attempt to send.

Created by

User ID of person who created this entry.

Confirm Delete of Job Start after Send Attempt Entry (mask DJSS)

In this panel, you are ask to confirm that you really want to delete this displayed JobStart after SendAttempt entry you selected in the previous panel.

Functionkey **<F1>** :	Display help information.
Functionkey **<F3>** :	Cancel delete request; the displayed entry will remain active. Leave display and return to previous panel.
Functionkey **<F5>** :	Confirm delete request; the displayed entry will be deleted

DATE	Current date
TIME	Current time
CmndNbr.	Internal number of this entry.
Remote Station	Station ID of receiver.
User ID of sender	User ID of sender. Name or pattern of user ID of sender.
virtual data setname	(Virtual) name of outgoing data set.
Attempts	Specifies amount of send-attempts after which the specified job should be started.
Job	Name of data set that can be started as a batch job, to specify the following process.

© gedas deutschland GmbH / Competence Center EDI/rvs®

Comment Brief comment describing the purpose of this entry (used for display only).

Created by user User ID of person who created this entry.

Creation date Date and time when this entry was created.

Last used date Date and time when this entry was last used to detremine the actions that occured when a data set was attempt to send.

Update Job Start after Send Attempt Entry (mask UJSS)

In this panel, you can update JobStart after SendAttempt entry. In a JobStart after SendAttempt entry, a user may specify what actions should be taken, when a data set has been attempt to send. This mask is similar to the mask `Create Job Start after Send Attempt Entry`, CJSS.

Functionkey **<F1>** : Display help information.

Functionkey **<F3>** : Terminate action and return to previous panel.

Key **<ENTER>** : If all fields are specified the entry will be updated.

1. STATION ID Station ID of receiver.

This is one of the selection criteria and so you will not be able to change this entry.

2. USER User ID of sender.

This is one of the selection criteria and so you will not be able to change this entry.

3. **DSNAME** (Virtual) name of outgoing data set.

This is one of the selection criteria and so you will not be able to change this entry.

4. **ATTEMPTS** Number of send-attempts before starting the specified job.

This is one of the selection criteria and so you will not be able to change this entry.

5. **JOB** Name of data set that can be started as a batch job, to specify the following process.

e.g.

- ____**.bat** on MS DOS or Windows NT
- ____**.CNTL** on MVS
- ____**.cmd** on OS2

On AS/400 the job will be started with the OS/400 system command: `SBMDBJOB`.

6. **COMMENT** Brief comment describing the purpose of this entry (used for display only).

© gedas deutschland GmbH / Competence Center EDI/rvs®

3.5.3 User List

This chapter describes how you can create, display, confirm and update user entries.

List of User Entries (mask USRL)

```
------------------------------------------------------------------------------
|         R V S  -  DISPLAY / UPDATE / DELETE USER
------------------------------------------------------------------------------
                                                Date = 2000/03/29 Time = 09:58

   Optionen: S - display   U - update   D - delete

  User-ID    full Name                          Authorization    Language
------------------------------------------------------------------------------
| |bwa                                         Administrator    Deutsch
  pw         ?                                  Administrator    Deutsch




------------------------------------------------------------------------------
   F1 = HELP, F2 = NEW ENTRY, F3/ESC = EXIT, F5 = REFRESH
Row 1 .. 2 - End of table reached
```

This panel displays a list of rvs® user entries.

Use line command **<S>**, **<U>**, or **<D>** to get more information about the entry, update, or delete it, respectively.

Functionkey **<F1>** : Display help information.

Functionkey **<F2>** : Display a mask to add a new entry.

Functionkey **<F3>** : Ignore any selection and leave display.

Functionkey **<F7>** : Page up.

Functionkey **<F8>** : Page down.

Key **<ENTER>** : Confirm selection.

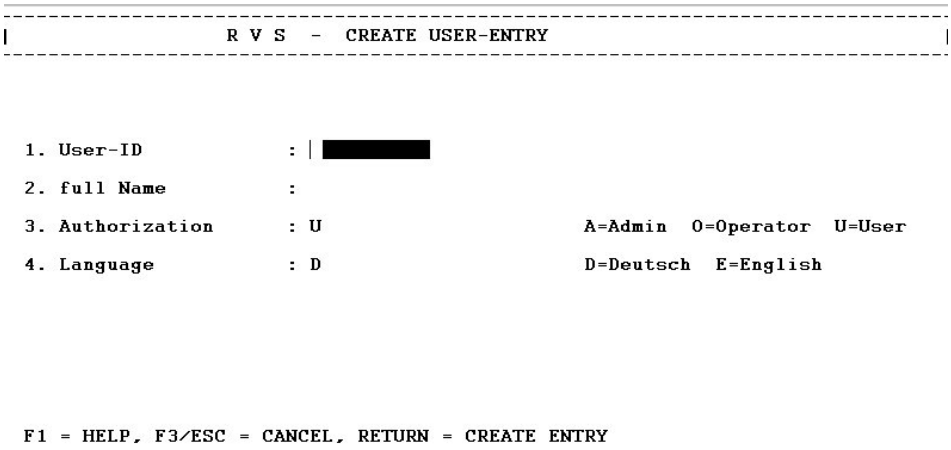
DATE Current date

TIME Current time

Specify option	Enter one of these options to the left of one of the displayed user entries: • **S**: Display the entry. • **U**: Update the entry. • **D**: Delete the entry.
UID	User ID.
full Name	full Name or description of the user.
Authorization	Authorization to use rvs®.
Language	Language to display panels and messages for this user.

© gedas deutschland GmbH / Competence Center EDI/rvs®

Create User Entry (mask CUSR)


R V S - CREATE USER-ENTRY
1. User-ID :
2. full Name :
3. Authorization : U A=Admin O=Operator U=User
4. Language : D D=Deutsch E=English
F1 = HELP, F3/ESC = CANCEL, RETURN = CREATE ENTRY

In this panel you can create a rvs® User Entry.

In a user entry you can grant authorizations to work with rvs® . Furthermore, you can specify the language to display masks and messages for the specified user.

Functionkey **<F1>** : Display help information.

Functionkey **<F3>** : No entry will be created; leave the display and return to the previous one.

Key **<ENTER>** : If all necessary fields are filled the entry will be created.

User ID User ID which identifies the user on the system.

full Name full Name or description of the user.

Authorization Authorization to use rvs®.

This authorization has no affect on the system authorization of this user.

- **A** - Administrator
- **O** - Operator
- **U** - User

Language Language to display panels and messages for this user.

- **D** - German
- **E** - English

Password Password for this user.

Display User Entry (mask SUSR)

This panel displays details about the rvs® User Entry you selected in the previous panel.

Functionkey **<F1>** : Display help information.

Functionkey **<F3>** : leave the display and return to the previous one.

DATE Current date.

TIME Current time.

User ID User ID which identifies the user on the system.

full Name full Name or description of the user.

© gedas deutschland GmbH / Competence Center EDI/rvs®

Authorization Authorization to use rvs.

This authorization has no affect on the system-authorization of this user.

- **A** - Administrator
- **O** - Operator
- **U** - User

Language Language to display panels and messages for this user.

- **D** - German
- **E** - English

Password Password for this user.

Confirm Delete of User Entry (mask DUSR)

In this panel, you are asked to confirm that you really want to delete the displayed user entry.

Functionkey **<F1>** : Display help information.

Functionkey **<F3>** : Cancel delete request; the displayed entry will remain active.

Leave display and return to previous panel.

Functionkey **<F5>** : Confirm delete request; the displayed entry will be deleted

DATE Current date.

TIME Current time.

User ID User ID which identifies the user on the system.

full Name full Name or description of the user.

Authorization Authorization to use rvs®.

Language Language to display panels and messages for this user.

Password Password for this user.

Update User Entry (mask UUSR)

In this panel you can update a rvs® User Entry. This mask is similar to the mask `Create User Entry`, CUSR.

In a user entry you can grant authorizations to work with rvs® . Furthermore, you can specify the language to display masks and messages for the specified user.

Functionkey **<F1>** : Display help information.

Functionkey **<F3>** : No entry will be created; leave display the display and return to the previous one.

Key **<ENTER>** : If all necessary fields are filled the entry will be updated.

User ID User ID which identifies the user on the system.

full Name full Name or description of the user.

© gedas deutschland GmbH / Competence Center EDI/rvs®

Authorization

Authorization to use rvs®.

This authorization has no affect on the system-authorization of this user.

- **A** - Administrator
- **O** - Operator
- **U** - User

Language

Language to display panels and messages for this user.

- **D** - German
- **E** – English

Password

Password for this user.

3.5.4 Information about rvs®

Information (mask INFO)

OPTION Select one of these options:

1. **RVS NEWS** (if available) may be listed, using this option (not yet implemented).
2. General Informations about rvs® may be listed (not yet implemented).
3. Lists all nodes that can be reached from the local station (either directly or indirectly).
4. Print rvs® manual (not yet implemented).

Information for options 1 and 2 is provided and maintained by your local rvs® administrator.

copies Specify how many copies you want to print of rvs® User Manual

© gedas deutschland GmbH / Competence Center EDI/rvs®

3.6 Interactive Sending and Receiving for Windows NT

This chapter describes how to send and receive data sets with rvs® if you are using an ineractive application. First, it will be explained how you can create send requests. The next chapter describes how you can display sent and received transmissions and the last chapter shows how you can operate with send and receive data sets.

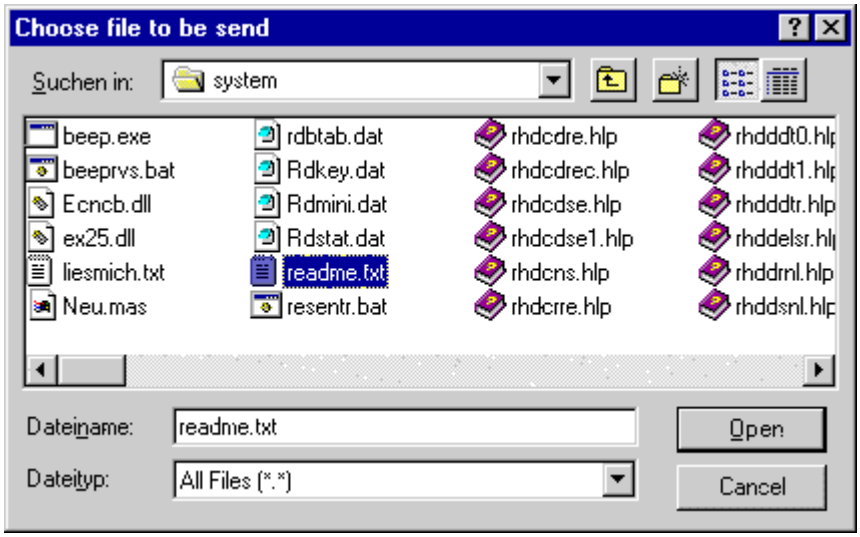
3.6.1 Create Send Requests

This is how you send a file to a business partner:

- Execute the menu command `Edit` → `New Send Entry...`.

You can also execute this command by using **F2** or by pressing the push button **New Send Entry**.

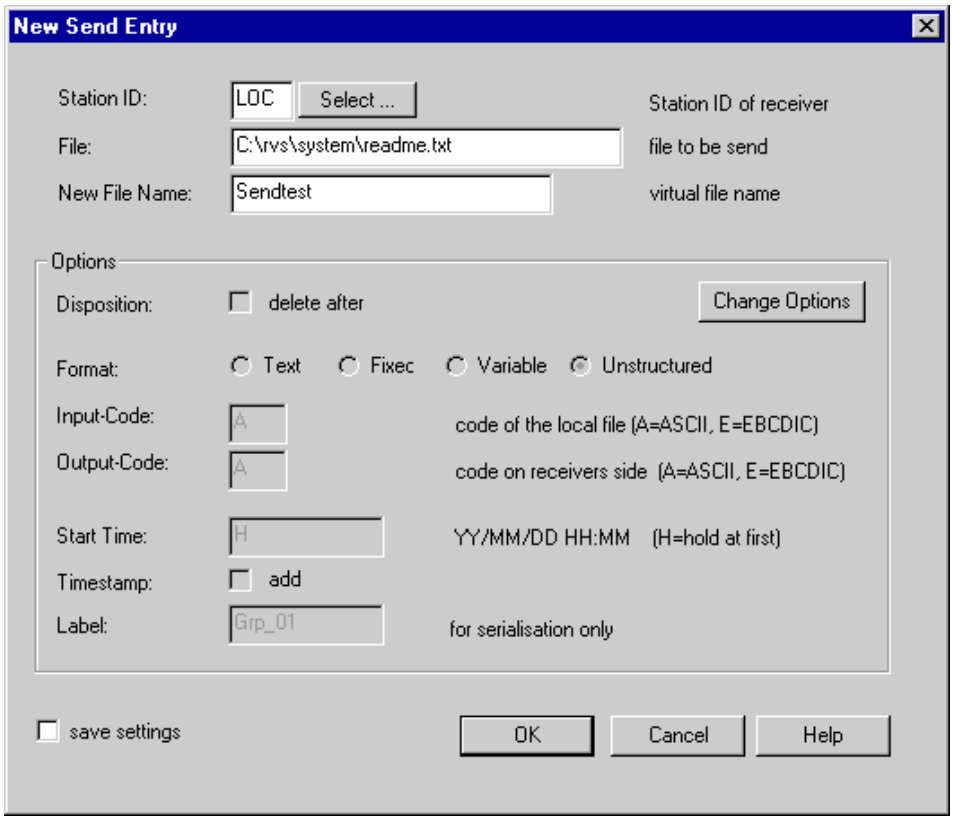
The `Choose file to be sent` dialog box will open.



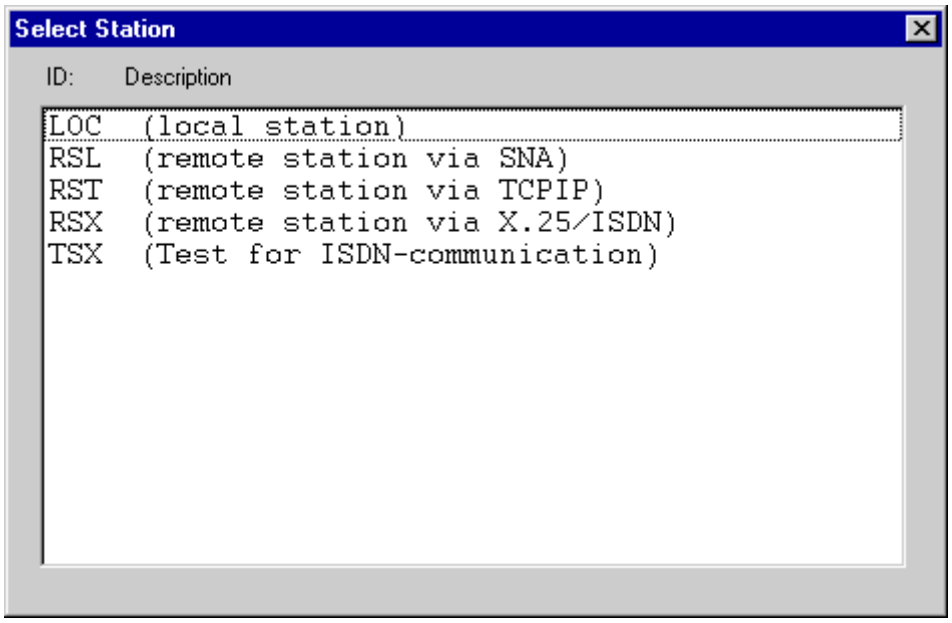
- Select the file which you would like to send to your partner and confirm your selection with **Open**.

Example: `c:\rvs\system\readme.txt`

The `New Send Entry` dialog window then opens and displays the file name selected in the `File` field.



- Manually enter the station code for the receiving station in the `Station ID` field. or:
- Use **Select...** to open the `Select Station selection` window and select the station code.



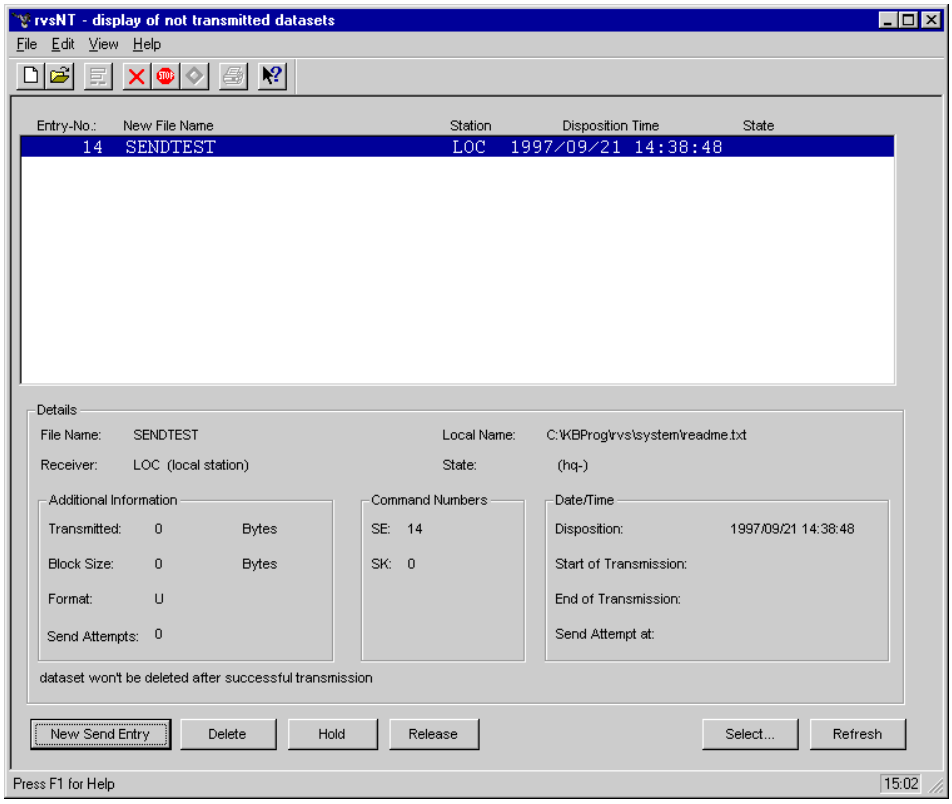
© gedas deutschland GmbH / Competence Center EDI/rvs®

Example: LOC

LOC is the standard setting.

- If you want to send the file to your partner under a different name, then enter the new file name into the `New File Name` field.
- Confirm the entries by pressing **OK**.

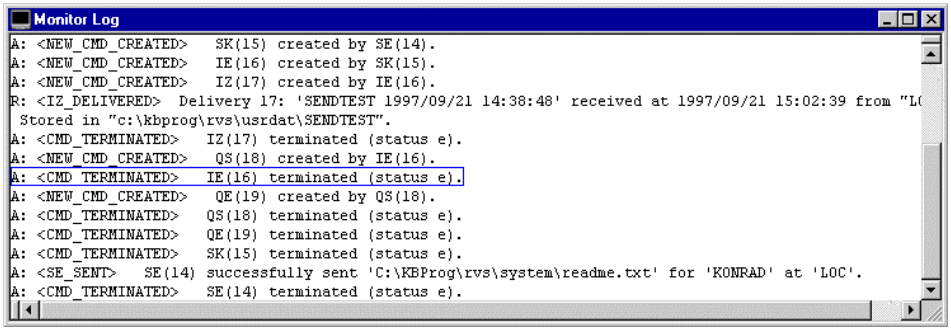
rvsNT then writes the send entry for transmission of the file to the rvs database. The send order has been completed. rvsNT displays the order in the list of files which have not yet been sent.



The rvs® Monitor reads the send order at the next read cycle and executes it:

In the example the rvs® Monitor recognizes through the station code LOC that its own station is the target and copies the file into the `c:\rvs\usrdat` directory.

NOTE: You can view the activities of the rvs® Monitor in the rvsNT Administrator in the `Monitor Log` window:



3.6.2 Display Send Requests and Received Transmission

Display files sent

You can choose between two lists to display the send orders:

Not yet completed: The list of send orders which have not yet been completed displays those files for which you have chosen to hold transmission or which are waiting in the queue of files to be transmitted.

- Execute the menu `command View | Send files` in order to display the send orders which have not yet been completed.

Completed: The list of completed send orders displays those files which have been successfully transmitted and acknowledged by the recipients.

- Execute the menu command View | Completely sent files in order to display the list of send orders which have been completed.

Update display

If you want to update the list display, do the following:

- Click the **update** button or switch to a different display mode and then back again.

© gedas deutschland GmbH / Competence Center EDI/rvs®

Display received files

You can choose between two lists to display the received files:

File being received: The list of files being received displays those files which have not yet been completely transmitted. Your station and the partner station are presently transmitting the files. This is how you display the files announced:

- Execute the menu command `View → Receive files`.

File completely received: The list of files which have been completely received displays those files which have been successfully transmitted to your station. At this stage your station has acknowledged reception. This is how you display completed send orders:

- Execute the menu command `View → Completely received files`.

Update display

If you want to update the list display, then do the following:

- Click the **update** button or switch to a different display mode and then back again.

3.6.3 Operate Send Requests and Received Transmission

Hold send order

Since rvsNT operates asynchronously there is always a certain time span between your send order and file transmission. This depends on the rvs® Monitor capacity and settings. During this time you can still put a hold on a send order and modify it.

This is how you hold one or more send orders:

- Use the mouse to click on a single send order.

NOTE: To find out how to limit the display to the send orders placed on hold or certain file types, please refer to the chapter "".

- Click the **Hold** button or execute menu command `Edit → Hold`.

Modifying a send order

This is how you modify a send order:

- Use the mouse to click the send order that you want to modify.
- Execute the menu command `Edit` → `Change`.

The `Change Send Entry` dialog window then opens. In this window you can modify all the fields for the send order.

- Carry out the changes desired
- Confirm your entries with **OK**.

The send order has been modified. rvsNT displays the modified order in the list of files which have not yet been sent.

Temporarily hold

If you want to create a send order and then release it later, you should carry out the following steps to put a temporary hold on it:

- Click **Change Options** in the `New Send Entry` dialog window.
- Enter the letter **H** in the `Start Time` field.
- Confirm the entry by pressing the button **OK**.

Releasing a send order

If you have put a hold on a send order, you can also release it for subsequent execution.

This is how you release one or more orders for execution:

- Use the mouse to click on a single send order.

NOTE: To find out how to limit the display to the send orders placed on hold or certain file types, please refer to the chapter "Select display criteria".

- Release the orders selected with **Release** or the `Edit` → `Release` menu command.

© gedas deutschland GmbH / Competence Center EDI/rvs®

Restart send order

rvsNT repeats an unsuccessful transmission attempt in accordance with the time specified by means of the **DTCONNnn** parameter. You can avoid this waiting period and immediately start a new attempt at transmission. The rvs® Monitor then treats the send order as though you had created it for the first time.

This is how you restart one or more send orders:

- Use the mouse to click on a single send order.
- Click the **Hold** button or execute menu command `Edit` → `Hold`.
- Release the orders selected with **Release** or the `Edit` → `Release` menu command.

Remove send order

You can remove a send order from the database. If you remove a send order which has the status `forwardable` or `held`, then your partner will not receive the file which was intended for him.

A send order that has already been completed remains executed. It is not possible to revoke the order. You can only remove the entry from the rvs® database. This may make sense in some cases in order to improve management of transmission activities.

NOTE: The rvs® Monitor constantly purges the rvs® database of send entries which have been completed since a particular period of time (**KEEPDAYS** parameter). Thus you do not have to remove the send orders in order to clean the rvs® database.

This is how you remove one or more send orders from the order list:

- Use the mouse to click on a single send order which you want to remove.

NOTE: To find out how to limit the display to the send orders placed on hold or certain file types, please refer to the chapter "Select display criteria".

- Click the **Delete** button or execute menu command `Edit` → `Delete`.

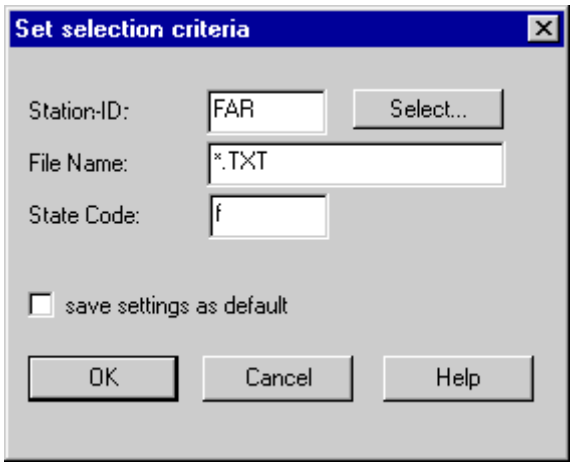
3.6.4 Select display criteria

You can select certain kinds of send orders for display in the entry list. You can use the station code, file name and status of the orders as selection criteria.

This is how you set the display criteria for send orders:

Execute the menu command `View` → `Select`.

The `Set selection criteria` dialog window then opens.



- Enter the station code for the station(s) to be selected.

or:

- Use **Select...** to open the `Select Station selection` window and select the station code.
- Enter the criterion for the files to be selected in the `File Name` field.
- Enter the status code to be selected in the `State Code` field.

This entry is optional.

NOTE: You can use the wildcard characters ***** and **?** or a combination of ***** and **?** for entering selection criteria.

***** stands for any - or even empty - character string

? stands for one character of any value

© gedas deutschland GmbH / Competence Center EDI/rvs®

Use one of the following characters as the status code:

-	no event
q	queued, not yet processed by rvs® Monitor
f	forwardable, waiting for transmission process
p	pending, waiting for completion
e	ended
a	active
i	in transit, transmitter currently processing order
d	deleted
h	held by the system or operator
s	traffic with the destination suspended at present

- If you want to repeatedly use this setting then activate the `Save Setting as Standard` selection box.
- Confirm the entry by pressing the button **OK**.

3.7 Interactive Administration for Windows NT

This chapter describes how to administrate resident receive entries, jobs after send attempts and user lists.

You must call the application `rvsmgr` if you want to work interactively with rvs®.

3.7.1 Resident Receive Entries

When a file is received the rvs® Monitor stores it in the directory `c:\rvs\usrdat`. Then it logs reception in the rvsNT Administrator and in the rvs® log book.

You can trigger other actions when receiving files by generating resident receive entries (RE) in the rvs® Database. A RE tells the Monitor to start a program when a particular file or selection of files has arrived.

Example:

When a file with the extension TXT (*.TXT) arrives, the batch program `copy.bat` should start (see the following).

This batch program copies the received file from the user data directory c:\rvs\usrdat into the directory `d:\temp`.

```
copy ?DSN? d:\temp
```

DSN stands for DataSetName and states where the file was placed in the local file system. **?...?** are rvs replacement markers and rvs® replaces the character string **?DSN?** by the local file name of the received file.

Example:

```
c:\rvs\usrdat\konto1.txt or
c:\rvs\usrdat\konto2.txt etc.
```

Another example is to be found in the file:

`c:\rvs\system\ resentr.bat`.

© gedas deutschland GmbH / Competence Center EDI/rvs®

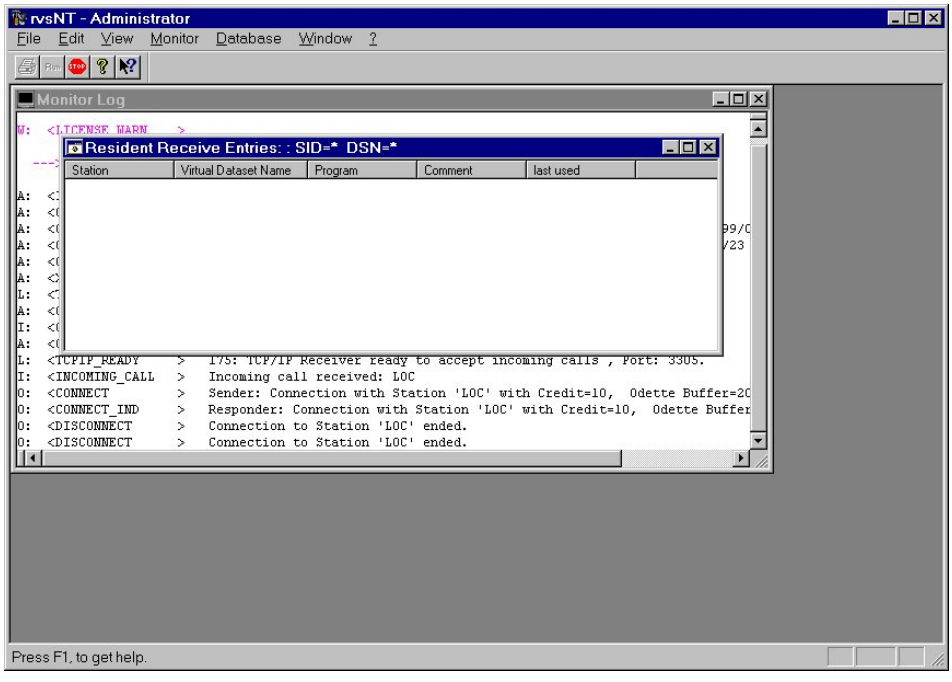
NOTE: No binary file may be specified as a batch program, it should be a readable and interpretable file. Ensure that your batch program is sensibly structured because it is not checked by rvs®.

Every version of an RE leaves a copy of the batch file, in which the replacement characters **?...?** are replaced by your actual values, in the directory `c:\rvs\temp`. The copies are stored under the name `re999999.bat`, whereby 999999 stands for a consecutive number. If the behaviour of a RE is incomprehensible to you, you should read in the last generated copy of the batch program (file `re999999.bat`) just which commands have been executed. You can find more information in section "How to work with rvs® Batch Interface and rvs® C-CAL Interface (rvscal())".

Display resident receive entry

If you would like to display the RE then use the `View` → `JobStart` → `Receive` menu command in the rvsNT Administrator.

The `Resident Receive Entries` window then opens. It displays a list of the previously created RE's. The list is initially empty.

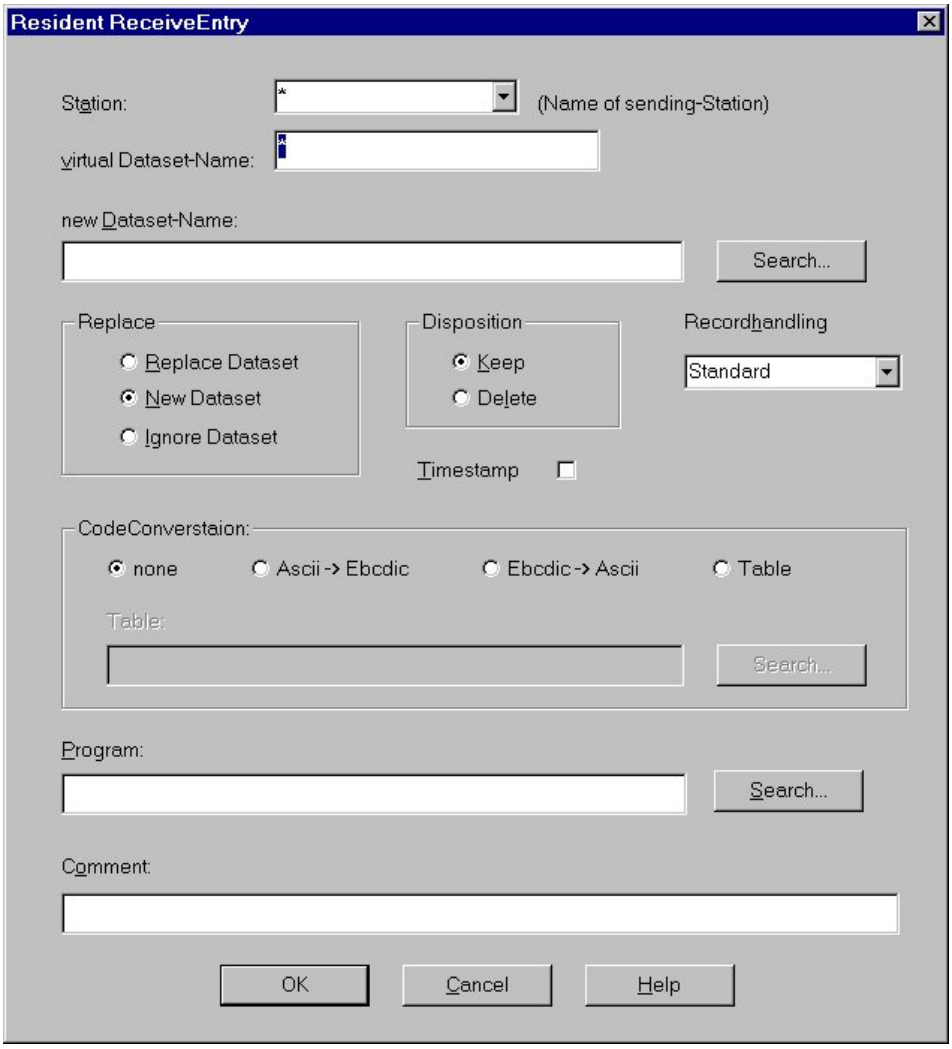


Create RE

If you want to start a program when a file arrives, do the following:

- Execute the menu command `Edit` → `Add`.

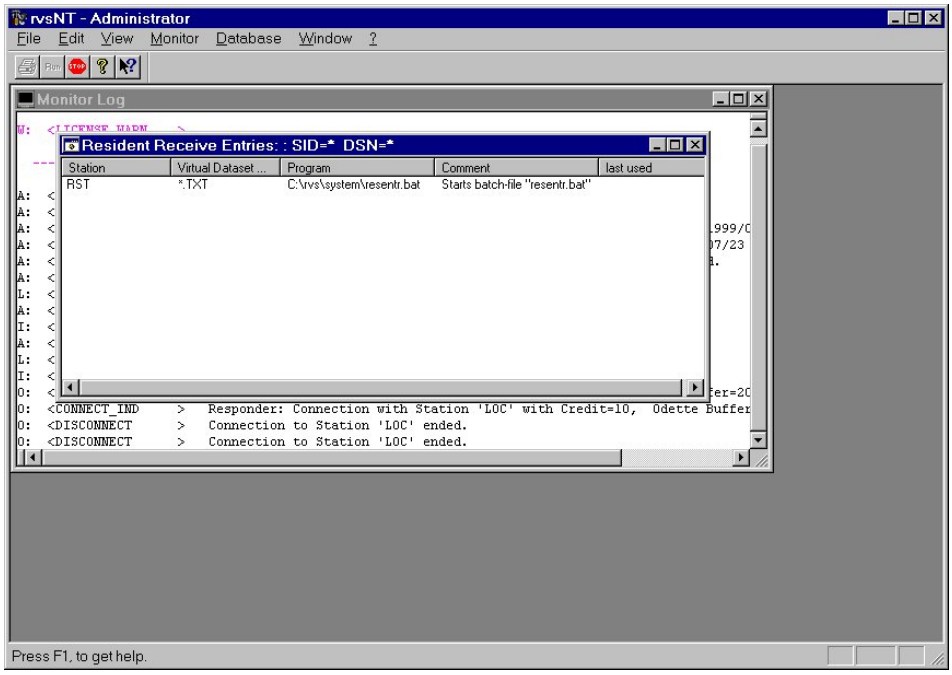
The `Resident Receive Entries` dialog window then opens.



- Enter the station from which you expect to receive the file. (The dialog field **Station** is pre-assigned with *****. This means that the RE is valid for the reception of files from all stations.)
- In the **virtual Dataset-Name** field enter the file or the file specification for which the RE is to apply. **Virtual Dataset-Name** is the file name that is used for the ODETTE transfer and has to be agreed upon by both sides.
- If you want to change the file name after it has arrived, enter the new file name in the field **new Dataset-Name**.

© gedas deutschland GmbH / Competence Center EDI/rvs®

- **Disposition**: determines whether the received file is deleted or kept after reception and execution of the RE.
- If you have decided in favour of **Keep disposition** and there is already a file with the same name in the user directory `c:\rvs\usrdat`, you may choose in the checkbox **Replace** whether you would like to replace the existing file, to create a new file with a distinct name (default) or to ignore the incoming file.
- Specify in the field **Recordhandling** whether the received file is to be stored as a text file with a line feed after every record. This specification only applies to files that are received in **Fixed** or **Variable** format. Select **Text** here if the line feed is to be switched on.
- In the field **CodeConversion** you may specify whether a code conversion is to take place. Select a special conversion table by means of **Search**. For standard conversion click **Ascii-Ebcdic** or **Ebcdic->Ascii**.
- Use **Search...** to select the batch program that should start when the file arrives.
- Confirm the entries by pressing **OK**.
  The rvsNT Administrator writes the new entry to the rvs® database and displays it in the list of RE's.



Modify RE

If you want to modify a RE:

- Select the entry to be modified in the `Resident Receive Entries` window.
- Execute the menu command `Edit` → `Edit`.

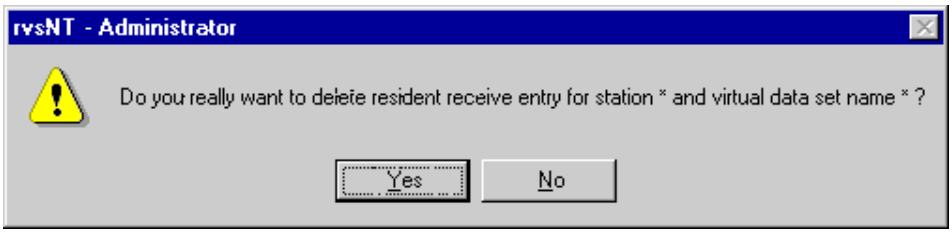
The `Resident Receive Entries` dialog window then opens and displays the properties of the entry.

- Process the entry properties and confirm your entries with **OK**.

Delete RE

If you want to delete a RE:

- Select the entry to be deleted in the `Resident Receive Entries` window.
- Execute the menu command `Edit` → `Remove`.
  rvsNT will then ask you to confirm your delete command.



- If you are sure that you want to delete the entry, then confirm the command with **Yes**. With **No** you cancel the operation without deleting.

© gedas deutschland GmbH / Competence Center EDI/rvs®

3.7.2 Job Start after Send Attempt

You can trigger actions, after rvsNT has attempted (with or without success) to send files by generating resident job start entries (JS) in the rvs® Database. A JS tells the rvs® Monitor to start a program after an attempt to send a particular file or a selection of files.

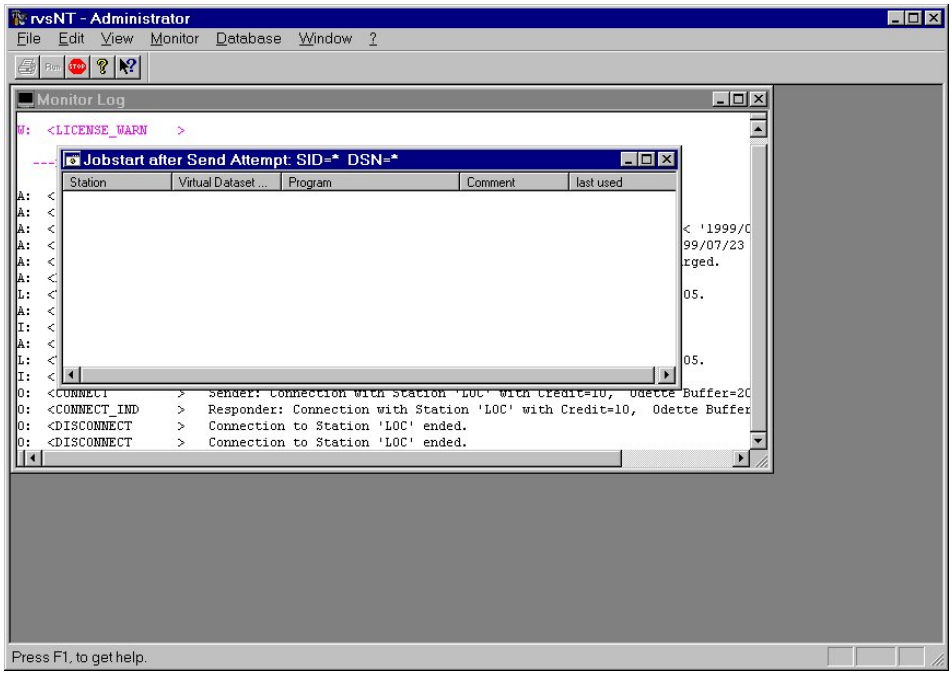
Example: The batch program `sendjob.bat` should start after the third unsuccessful attempt to send a file. This example is also to be found in the directory `c:\rvs\system`.

You can also use rvs® to switch parameters using the rvs® batch interface. The modified copies of the batch program also come into being here in the temp directory, exactly as is the case with RE's.

You can find more information in section "How to work with rvs® Batch Interface and rvs® C-CAL Interface (rvscal())".

Display job start after send attempt

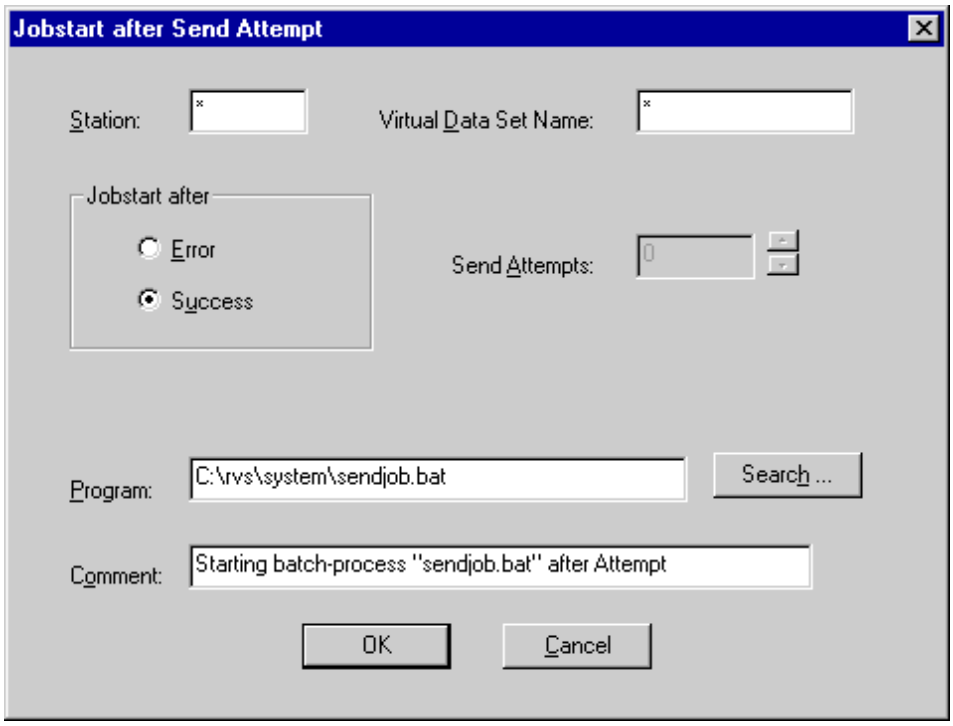
If you would like to display the JS then use the `View` → `JobStart` → `Send` menu command in the rvsNT Administrator. The `Jobstart after Send Attempt` window then opens. It displays a list of the previously created JS's. The list is initially empty.



Create JS

If you want to start a program after an attempt to send a file, do the following:

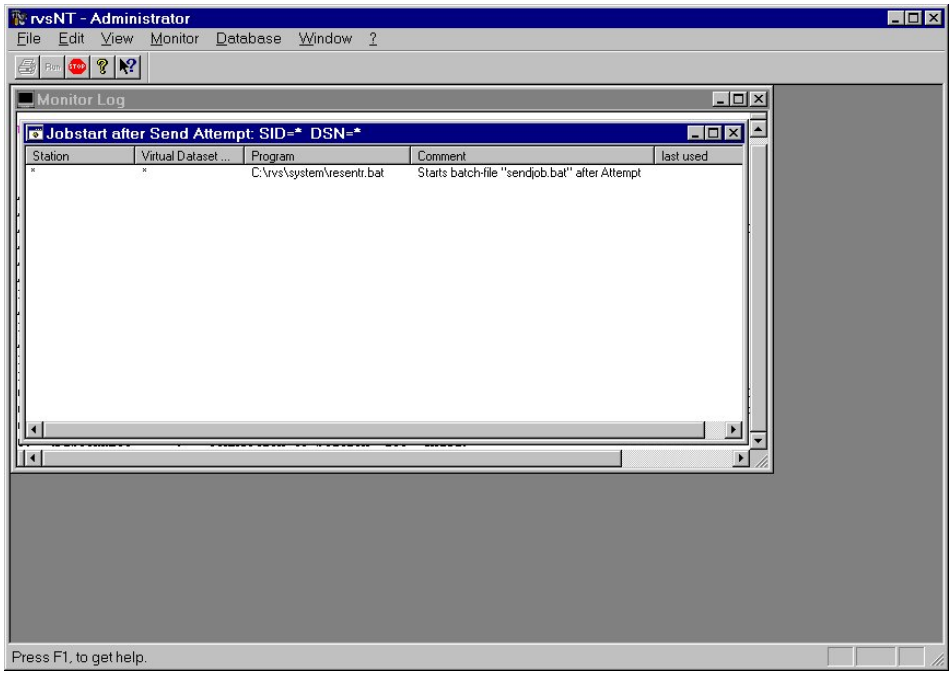
- Execute the menu command `View` → `Add`.
  The `Jobstart after Send Attempt` dialog window then opens.



- Enter the station you want to send the file to.
- In the **Virtual Data Set Name** field enter the file or the file specification for which the JS is to apply.
- Use **Search...** to select the batch program that should start after the attempt to send the file.
- If you only want the program to start in the event of an unsuccessful attempt, then select **Error** and under **Send Attempts** enter the number of unsuccessful attempts after which the program should start.
- Confirm the entries by pressing **OK**.

© gedas deutschland GmbH / Competence Center EDI/rvs®

The rvsNT Administrator writes the new entry t the rvs® Database and displays it in the list of JS's.



Modify JS

If you want to modify a JS:

- Select the entry to be modified in the `Jobstart after Send Attempt` window.
- Execute the menu command `Edit` → `Edit`.

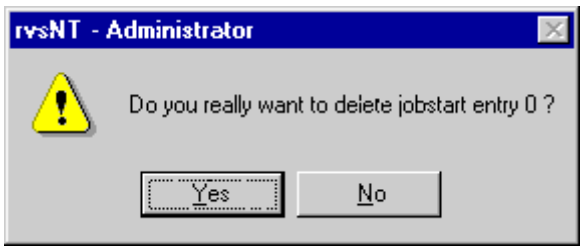
The `Jobstart after Send Attempt` dialog window then opens and displays the properties of the entry.

- Process the entry properties and confirm your entries with **OK**.

Delete JS

If you want to delete a `JS`:

- Select the entry to be deleted in the `Jobstart after Send Attempt` window.
- Execute the menu command `Edit` → `Remove`.
  rvsNT will then ask you to confirm your delete command.



- If you are sure you want to delete the entry, confirm the command with **Yes**. With **No** you cancel the operation without deleting the entry.

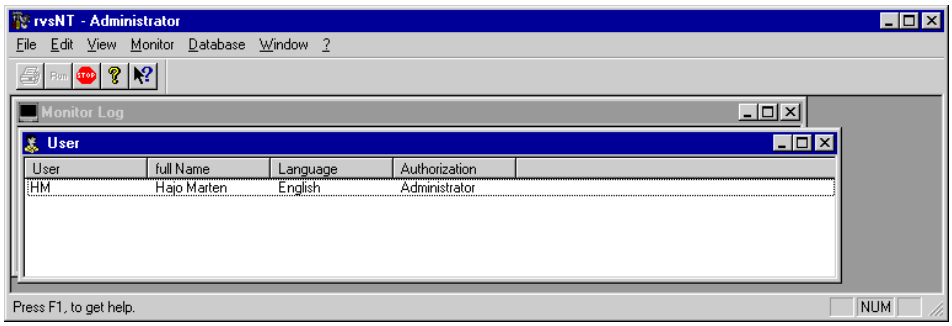
© gedas deutschland GmbH / Competence Center EDI/rvs®

3.7.3 User Entries

Only those users may work with rvsNT who are registered in the rvsNT database. In order to display or modify user data:

- Open the User Table in the rvsNT Administrator by using the `View` → `User` menu command.

The user table contains at least the table entry for your own data.



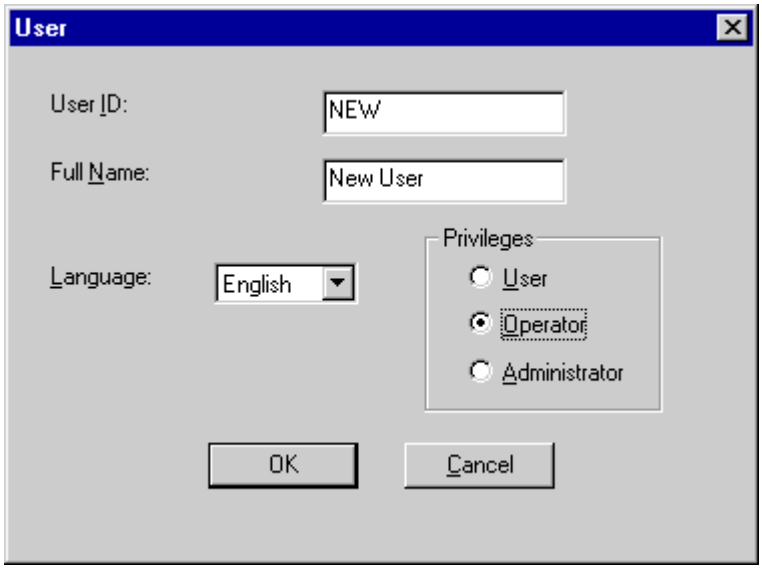
Sort

You can sort the table according to any column by using the mouse to click the column header.

New user

If you would like to set up a new user:

- Execute the menu command `Edit` → `Add`. The `User` dialog window then opens and displays the User Table parameters.



- Enter the user identification.
- Enter the user name.
- Select the language for screen dialog.
- Use the mouse to click on the user access authorization.
- Confirm the entries by pressing **OK**.

NOTE: A basic user may use the program rvsNT to send and receive files. A user with administrator rights may in addition to this use the program `rvsNT Administrator` to configure the rvs® system. An Operator has the same rights as an User. This type of user is reserved for future use.

User Modification

If you would like to modify an existing user:

- Select the user to be modified in the `User` window.
- Execute the menu command `Edit` → `Edit`.

The `User` dialog box will open.

You can now edit all of the user data parameters. You can select German or English in the language field. rvsNT will then display all the menus and messages in the selected language.

NOTE: If you want to modify your own user data then the entries will only take effect after restarting the rvsNT Administrator.

- Confirm the entries by pressing **OK**.

Save

If you set up or modify a user, use **OK** in order to save the modified parameters. If you delete the rvsNT database or reinstall rvsNT you will also delete all users. Subsequent initializing of the database only retains the administrator who has set up rvsNT.

In order to avoid losing your entries you can save the users and the job start entries or even the entire rvs® database and restore them later. If you choose to do so you can find more information in the Operator Manual section "Database Maintenance for Windows NT".

© gedas deutschland GmbH / Competence Center EDI/rvs®

4 How to work with rvs® Batch Interface and rvs® C-CAL Interface (rvscal())

This chapter describes how to start the two interfaces and which parameters you can use. Furthermore, a description of the corresponding global commands is given. The syntax of the commands is explained as well as the prototype of the function `rvscal()`.

Note: For all tasks (sending and receiving data files as well as administration of rvs®) you have to do with rvs® you can either

- call the general function `rvscal()` using string commands or
- use several dedicated functions

The string commands of `rvscal()` are identical to those useable for `rvsbat`.

On the other hand the dedicated functions are on a lower abstraction level thus providing more control and flexibility. They are described in section "How to Work with rvs® C-CAL Interface".

4.1 Start the rvs® Batch Interface

The batch interface can be invoked as specified below:

```
rvsbat [/c] [/e<env.dsn>] [/i<cmd input file>]
[/l<language>] [/q] [/t]
```

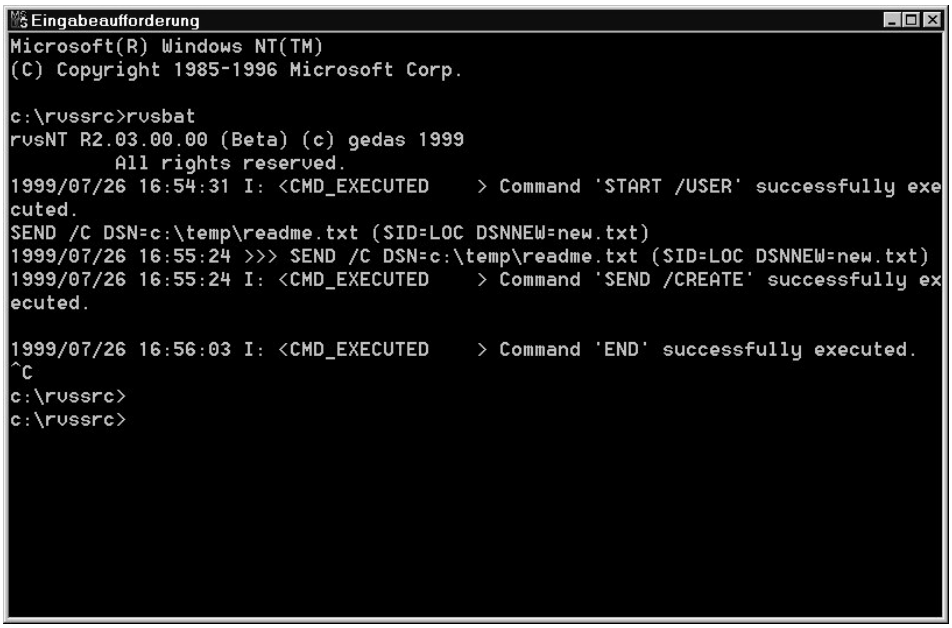
Note: the invocation and parameter passing mechanism are dependend on the target operating system.

The command line parameters have the following meaning:

- **/c**: continue with `rvsbat` after an error occured during execution of a utility command. By default, `rvsbat` will terminate after an error.
- **/e**: use non-default environment data set
- **/i**`cmdfile`: do not read commands from `stdin` but from `cmdfile`. The command input file may contain the following elements:
  - Comment lines (starting with *****)
  - Commands (may extend over several lines by specifying **+** as the last character in the line to be continued)

- **/l**`language`: use message language given by character language
- **/q** execute user commands in quiet mode, i.e. do not echo them to standard output; feedback about success or failure of the operation will still be provided.
- **/t**: use test mode

The following picture shows an example for **Wondows NT**:



Handling of Commands

Commands can be written into the <cmdfile> or can be entered when calling `rvsbat`.

In both cases, the syntax of the command strings is identical. An example is given in the picture below:

```
SEND /C DSN=c:\temp\readme.txt (SID=LOC
DSNNEW=new.txt)
```

For each command read from command input, a message is written to **stdout** which gives information about success or failure of processing the command.

If `rvsbat` is started without input of a command file, it can be breaked by **<STRG> C**.

© gedas deutschland GmbH / Competence Center EDI/rvs®

4.2 Use the C-CAL Interface

To use the C-CAL Interface it is necessary to link the `rvscal` library

- `rvscal.dll` for **Windows NT** (dynamic link)
- `rvscal.a` for **UNIX** systems (static link)
- `rvscal` for **AS/400** (service program)

Then the user defined C program which uses this library has to include the header file `rvscal.h` into the C source code.

4.2.1 Header File `rvscal.h`

This chapter describes how to define a macro and explains a prototype of `rvscal`.

Macro definitions

storage class definitions:

```
#if !defined _LITYPE_INCLUDED
typedef char            STRING ;
# if defined OS400
   typedef int             SINT;
# else
   typedef short           SINT;
# endif
#endif

#include "rixstd.h"

#ifdef NT
#  define PROCDEF extern  __declspec(dllimport)
#  define PROCKEYW


#elif  _AFXDLL
#  define PROCDEF  extern "C"
#  define PROCKEYW

#else
#  define PROCDEF
#  define PROCKEYW
#endif
```

return codes:

```
#define RVSCAL_OK                          0
#define RVSCAL_END_FETCH -                 1
#define RVSCAL_CLOSE_FAILED               -2
#define RVSCAL_OPEN_FAILED                -3
#define RVSCAL_ERROR_CREATESEND           -4
#define RVSCAL_PARAMETER_CHECK            -5
#define RVSCAL_DBERROR                    -7
#define RVSCAL_INVALID_NAME               12
#define RVSCAL_INVALID_SID                13
#define RVSCAL_RE_NOT_FOUND               14
#define RVSCAL_NEITHER_X_NOR_ISDN         15
#define RVSCAL_INVALID_DSN                16
#define RVSCAL_INVALID_JOB                17
#define RVSCAL_NOT_PRIVILEGED             18
#define RVSCAL_DUPLICATE_RE               19
#define RVSCAL_INVALID_UID                20
#define RVSCAL_INVALID_OWN_PRIV           21
#define RVSCAL_INVALID_USER               22
#define RVSCAL_DUPLICATE_USER             23
#define RVSCAL_NEIGHBOR_STATION           24
#define RVSCAL_RC_WAKE_FAILED             25
#define RVSCAL_WRITE_INTO_PIPE            26
#define RVSCAL_DSN_NOT_EXIST              27
#define RVSCAL_ENVIRONMENT_NOT_EXIST      28
#define RVSCAL_DUPLICATE_JS               29
#define RVSCAL_JS_NOT_FOUND               30
#define RVSCAL_ERROR_GETVERSION           31
#define RVSCAL_ERROR_MALLOC_INFOELEMENT   32
#define RVSCAL_ERROR_MALLOC_SIDELEMENT    33
#define RVSCAL_INTERNAL_ERROR             99

#define RVSCAL_NOCMD RVSCAL_END_FETCH
```


© gedas deutschland GmbH / Competence Center EDI/rvs®

others:

```
#define RVSCAL_L_ACCT           RVSCAL_L_USID
#define RVSCAL_L_C1             L_C1
#define RVSCAL_L_DBVERSION      L_DBVERSION
#define RVSCAL_L_DSN            255
#define RVSCAL_L_DT             L_DT
#define RVSCAL_L_JOBID          L_JOBID
#define RVSCAL_L_KS             L_KS
#define RVSCAL_L_ODETTEID       L_ODETTEID
#define RVSCAL_L_LANG           L_LANG
#define RVSCAL_L_LU62_NETID     L_LU62_NETID
#define RVSCAL_L_LU62_LUNAME    L_LU62_LUNAME
#define RVSCAL_L_LU62_TPNAME    L_LU62_TPNAME
#define RVSCAL_L_LU62_UID       L_LU62_UID
#define RVSCAL_L_LU62_PASSW     L_LU62_PASSW
#define RVSCAL_L_LU62_PROF      L_LU62_PROF
#define RVSCAL_L_LU62_MODE      L_LU62_MODE
#define RVSCAL_L_LU62_PIP_DAT   L_LU62_PIP_DAT
#define RVSCAL_L_LX_NAME        L_LX_NAME
#define RVSCAL_L_LX_VALUE       L_LX_VALUE
#define RVSCAL_L_LOGMSG         L_LOGMSG
#define RVSCAL_L_NETID          RVSCAL_L_ODETTEID
#define RVSCAL_L_ERRMSG  512
#define RVSCAL_L_ODETTEID       L_ODETTEID
#define RVSCAL_L_OFTP_EERP      L_OFTP_EERP
#define RVSCAL_L_OFTP_USERD     L_OFTP_USERD
#define RVSCAL_L_OPSW           L_OPSW
#define RVSCAL_L_PARM_NAME      L_PARM_NAME
#define RVSCAL_L_PARM_VAL       L_PARM_VAL
#define RVSCAL_L_PHONE          L_PHONE
#define RVSCAL_L_PIPEBUFFER     20000
#define RVSCAL_L_PRIORITY       L_PRIORITY
#define RVSCAL_L_RECMNT         L_RECMNT
#define RVSCAL_L_SEUSRLABEL     L_SEUSRLABEL
#define RVSCAL_L_STATID         L_STATID
#define RVSCAL_L_STATNAME       L_STATNAME
#define RVSCAL_L_TCPIP_ADDR     L_TCPIP_ADDR
#define RVSCAL_L_TCPIP_SEC      L_TCPIP_SEC
#define RVSCAL_L_USID           L_USID
#define RVSCAL_L_USERNAME       L_USERNAME
#define RVSCAL_L_VDSN           L_VDSN
#define RVSCAL_L_X25_ADDR       L_X25_ADDR
#define RVSCAL_L_X25_ALIAS      L_X25_ALIAS
#define RVSCAL_L_X25_CUG        L_X25_CUG
#define RVSCAL_L_X25_FAC        L_X25_FAC
#define RVSCAL_L_X25_LINK       L_X25_LINK
#define RVSCAL_L_X25_SUBADDR    L_X25_SUBADDR
#define RVSCAL_L_X25_USDTA      L_X25_USDTA
#define RVSCAL_L_X25_ISDNNO     L_X25_ISDNNO
#define RVSCAL_L_X25_TIMEOUT    L_X25_TIMEOUT

#define SID_START          " "
```

Prototype of function `rvscal`

```
int rvscal(char  s_cmd[],
   int  *p_l_cmd,
   char *p_c_msglvl,
   char  s_msg[],
   int  *p_l_msg,
   long *p_cmdid);
```

Note: parameter list has been designed in this way in order to be callable by other programming languages, e.g. FORTRAN, as well.

Description of Parameters

RETURNVALUE (int)

=**0**, if `rvscal` succeeded (although the message text may have been truncated), different from **0**, if any failure occured.

There are two types of error codes:

- errors detected by the command parser are due to an invalid command syntax. The error codes are in the range between 1 and 7. A detailed description is given in "Messages and Codes".
- Any other error occured while rvs® tries to execute the specified command. The message string returned via **S_MSG** contains a description of the error reason.

S_CMD (char [], input)

pointer to character array with command string; the array size must be at least ***P_L_CMD+1**

P_L_CMD (int *, input)

length of command string; **S_CMD** must be zero-terminated, if ***P_L_CMD** is larger than length of command string

P_C_MSGLVL (char *, output)

returns the message level; may be **I** (informal), **W** (warning), **E** (error), **S** (severe)

© gedas deutschland GmbH / Competence Center EDI/rvs®

S_MSG	(char [], update) pointer to a character string of length at least ***P_L_MSG+1** ; the character string receives a message from `rvscal` which gives information about the success of calling `rvscal`
P_L_MSG	(int *, update) on input, ***P_L_MSG+1** is the size of the buffer used to receive the message from `rvscal`; on output, ***P_L_MSG** gives the actual length of the message text. This value does not include the zero-terminator, but **S_MSG** is zero-terminated.
P_CMDID	(long *, output) the **CMDID** (command number) the command which was processed (see table below).

The following commands (`s_cmd`) are supported:

`start`	start rvs® utility session
`end`	terminate session
`send`	create, modify (hold or release) or delete a send request
`print`	print documentation about rvs®
`receiver`	delete a single receiver from send request
`resentr`	create, update or delete a resident receive entry
`sendjob`	create, update or delete a entry for starting jobs after send attempts
`user`	create, modify or delete entry in rvs® user table
`activate`	activate a rvs® station
`modst`	modify station table (read a station table file)
`delst`	delete a station table entry
`listparm`	list a rvs® parameter
`setparm`	set a rvs® parameter

The **CMDID** returned by `rvscal` has the following meaning depending on the command passed to `rvscal`:

Command	meaning of CMDID
`SEND /CREATE`	number of created send entry
`SEND /DELETE`	number of send entry being deleted[1]
`SEND /HOLD`	number of send entry being held (see footnote below)
`SEND /RELEASE`	number of send entry being released (see footnote below)
`RESENTR /CREATE`	created resident receive entry
`RESENTR /DELETE`	deleted resident receive entry
`RESENTR /UPDATE`	number of new resident receive entry
`SENDJOB /CREATE`	created jobstart after send attempt entry
`SENDJOB /DELETE`	deleted jobstart after send attempt entry
`SENDJOB /UPDATE`	number of new jobstart after send attempt entry
`RECEIVER /DELETE`	number of corresponding send entry
`LISTPARM`	not equal **0**: invalid parameter
`SETPARM`	not equal **0**: invalid parameter
`MODST`	not equal **0**: error occured while loading station table
`any other command`	**0**

4.2.2 Description of Commands

This section describes the syntax and the set of valid commands used either by the rvs® C-CAL Interface (`rvscal`) or the rvs® batch interface (`rvsbat`).

The examples given below, use two syntactical extensions only available with the batch interface:

- comment lines start with a ***** in column **1**,
- commands can be continued in the next line by specifying a **+** as the last character of the line to be continued

[1] If the send entry could not be identified uniquely by the specified parameters, the number of the first matching command is returned.

© gedas deutschland GmbH / Competence Center EDI/rvs®

4.2.3 Syntax of Command Strings

A utility command must follow the syntax rules specified below:

- It consists of
  - a command verb,
  - an optional command qualifier like `/CREATE, /DELETE`, etc. The qualifier may start either with **/** or **-** and may be abbreviated to a single letter
  - values of non-repeatable parameters specified as <parameter name>=<parameter value> . There must not be blanks around the = sign.
  - values of repeatable parameters (`send` command only); they follow the same syntax rules as those for non-repeatable parameters. A group of repeatable parameters is enclosed in parentheses. There may be arbitrary many groups of repeatable parameters.
- Command verb, qualifiers and parameter names may be specified in uppercase, lowercase, or mixed case.
- Parameter values are converted to uppercase, if not protected by single or double quotes.
  If a parameter value is to contain the protecting character itself, the protecting character must be specified twice. E.g. the following parameter specifications are equivalent:

  ```
  PARM=' "test string" ' or parm="""test
  string"""
  ```

  Non-protected parameter values may contain any alphanumeric character. They may not contain a set of special characters, e.g. blanks, single or double quotes, and parentheses.
- Different parameter specifications must be delimited by at least one blank.

4.2.4 Command `Start`

Function:

- Start a session using rvs® utilities
- Open rvs® database,
- Check whether current user is allowed to use the rvs® utilities
- `start /USER` is issued implicitely by the C-CAL Interface, if not issued explicitely. It is always performed implicitely when the batch interface starts execution.

Qualifiers:

`/USER` (default) start session for rvs® user

Parameters:

RVSENV (optional) name of rvs® environment data set

Examples:

Use environment data set in (rvs400 library) **RVS_NEW/DAT(RVSENV)**

```
START /USER RVSENV="RVS_NEW/DAT(RVSENV)"
```


© gedas deutschland GmbH / Competence Center EDI/rvs®

4.2.5 Command End

Function:

- Terminate session using rvs® utilities
- Close rvs® database
- END must be issued when the C-CAL Interface is used. It is performed implicitely by the batch interface.

Qualifiers:

none

Parameters:

none

Examples:

```
END
```

4.2.6 Command Send

Function:

- Create
- modify or
- delete

a send request to send a local data set to another rvs® node

Qualifiers:

`/CREATE`	(default) create a send request
`/DELETE`	delete a send request
`/HOLD`	set pending send request into `held'-state
`/RELEASE`	release send request, which has been put into `held'-prior

`SEND /CREATE` parameters:

DSN	(required) name of local data set to be sent
CODEIN	(optional) code of local data set (**A**=ASCII, **E**=EBCDIC); default: local code

DISP	(optional) disposition for local data set after send request has ended successfully: **K**=keep (default), **D**=delete
FORMAT	(optional) format used for transfer via ODETTE: **T**=text, **U**=unstructured (binary), **F**=fixed, **V**=variable record length; default: record format of local data set (**U** for rvsNT, rvsX and rvs2, **F** for rvs400)
ACCOUNT	accounting number or code
INITTIME	(optional) earliest time when send request may be performed; may be: **H**=hold, **N**=now (default), or an explicit time in the format **YYYY/MM/DD HH:MM**
LABEL	(optional) user label (up to 20 characters) used to serialize on a preceeding send request (if **SERIAL**=**Y**) or which can be used for serialization by a subsequent send request
SERIAL	(optional) if set to **Y** (=yes), the send requests will sent in the order you have created them (see also **LABEL**). The next request will only be sent if the previous is completely finished.

Repeatable Parameters:

Each group of repeatable parameters defines one receiver:

SID	(required) station ID of receiver
UID	(optional) user ID of receiver; if omitted or empty string: system at remote station
DSNNEW	(optional) virtual data set name used for transfer; for transfer to MVS host, this must be a valid MVS data set name (acceptable to RACF) default: virtual data set name is constructed from name of local data set
CODEOUT	(optional) desired code of data set at receiver
TSTAMP	(optional) determines, whether data set is to be timestamped at receiver. Currently, this request cannot be passed to or via an MVS host (see "Data Set Names / Time Stamping" for more information).

© gedas deutschland GmbH / Competence Center EDI/rvs®

`SEND /DELETE, /HOLD, /RELEASE` parameters:

DSN (optional) name of local data set to be sent

SID (optional) station ID of receiver

UID (optional) user ID of receiver; defaults to empty string, i.e. the remote system

CMDID (optional) unique command number of send request.

The send entry to be processed can be identified either by specifying **DSN**, **SID**, and **UID**, or by the unique command number **CMDID** of the send request.

If no parameter has been given or if more than one send request matches the specified parameters, `rvscal` does not modify any send request and returns with an error.

The following examples demonstrate, how commands may be specified in a data set that is used as input file to `rvsbat`, including use of continuation lines (the previous line ends with **+**) and comments (***** in column 1).

Examples:

```
*
*------ use all parameters
*
SEND /C DSN=C:/RVS/LPDBI.C +
SERIAL=n LABEL=l1 inittime=NOW +
CODEIN=A FORMAT=T DISP=d +
+
    (SID=st1 UID=user1 CODEOUT=e TSTAMP=n
DSNNEW=dsnnew1) +
    (SID=st2 UID=user2 CODEOUT=a TSTAMP=y
DSNNEW=dsnnew2)
*
*------ use defaults and serialize on LABEL
*
SEND    DSN=LPDBI.C +
SERIAL=y LABEL=l1 inittime=HOLD +
CODEIN=E FORMAT=U DISP=k +
+
     (SID=st1)
*
*------ serialize again on LABEL
*
SEND    DSN=rpu.c +
SERIAL=y LABEL=l1 inittime='1991/07/01 10:35' +
FORMAT=V +
+
     (SID=st1)
*
*------ serialize on data set (without specifying
the full data set name)
*
SEND    DSN=lpdbi.c +
SERIAL=y +
FORMAT=f +
+
     (SID=st1)
*
*------ serialize again on data set (and default
for FORMAT)
*
SEND    DSN=lpdbi.c +
SERIAL=y +
+
     (SID=st1)
*
*------ delete SEND request (unique)
*
SEND /D DSN=rpu.c (SID=st1)
*
```


© gedas deutschland GmbH / Competence Center EDI/rvs®

4.2.7 Command RESENTR

Function:

Create, update, or delete a resident receive entry.

Qualifiers:

/CREATE	(default) create resident receive entry
/UPDATE	update resident receive entry
/DELETE	delete resident receive entry

Parameters:

DSN	(required) virtual name of incoming data set
SID	(required) station ID of sender
ACCOUNT	(optional) accounting number or code
COMMENT	(optional) comment describing action of resident receive entry (up to 50 characters); default: empty string
DSNNEW	(optional) local name to be used for received data set; default: local data set name is constructed from virtual data set name
JOB	(optional) name of data set containing a job to be submitted after data has been received; if specified, the data set must exist. default: no job will be submitted.
DISP	(optional) determines what should be done with the incoming data set after processing has been completed: **D**=delete; **K**=keep default: **K**
REPLACE	(optional) determines action of rvs® when local data set does already exist and **DISP**=**K**; may be **R** (replace existing data set), **N** (create new data set with unique name), or **I** (ignore incoming data set) default: **N**
TSTAMP	(optional) may be **Y**=yes or **N**=no; tells rvs, whether the data set name is to be timestamped to make it unique, when the data set is received; default: **N**

Examples:

```
*
*---------- use all parameters
*
RESENTR /C DSN=incoming1    SID=st2 +
   DSNNEW=local.dsn REPLACE=n DISP=k +
   JOB=/home/rvs/bin/rcv.sh  COMMENT='This is a
test RE'
*
*---------- use defaults
*
RESENTR /C DSN=incoming2    SID=st2 +
   REPLACE=i        DISP=d
*
*---------- no UID, DISP
*
RESENTR /C DSN=incoming3    SID=st2 +
   REPLACE=r
*
*---------- delete RESENTR
*
RESENTR /D DSN=incoming3    SID=st2
*
*---------- update RESENTR
*
RESENTR /U DSN=incoming2    SID=st2 +
   REPLACE=n        JOB=/home/rvs/bin/rcv.sh
```


© gedas deutschland GmbH / Competence Center EDI/rvs®

4.2.8 Command SENDJOB

Function:

Create, update, or delete a jobstart after send attempt entry

Qualifiers:

/CREATE	(default) create jobstart after send attempt entry
/UPDATE	update jobstart after send attempt entry
/DELETE	delete jobstart after send attempt entry

Parameters:

VDSN	(required) virtual name of data set that is sent
SID	(required) station ID of receiver
ATTEMPTS	(optional) number of send attempts; determines in which case the job should start: value **0**: the job starts if data set has been transmitted successfully; value greater than **0**: determines the number of attempts to send the data set in vain after that the specified job should start default: **0**
JOB	(optional) name of data set containing a job to be submitted after data has been transmitted successfully or sending of data has been attempted in vain; if specified, the data set must exist
COMMENT	(optional) comment describing action of resident receive entry (up to 50 characters); default: empty string

Examples:

```
*
*---------- use all parameters
*
SENDJOB /C VDSN=sending1    SID=st2   ATTEMPTS=1
+
   JOB=/home/rvs/bin/send-fail1.sh COMMENT='This
is a test JS'
*
*---------- use defaults
*
SENDJOB /C VDSN=sending2    SID=st2
JOB=/home/rvs/bin/snd.sh
*
*---------- Job should start after data have been
transmitted successfully
*
SENDJOB /C VDSN=sending3    SID=st2
JOB=/home/rvs/bin/snd.sh
*
*---------- delete SENDJOB
*
SENDJOB /D VDSN=sending3    SID=st2
*
*---------- update SENDJOB
*
SENDJOB /U DSN=sending2    SID=st2
JOB=/home/rvs/bin/sendok.sh
```


© gedas deutschland GmbH / Competence Center EDI/rvs®

4.2.9 Command USER

Function:

- modify rvs® user table
- set user name
- set dialog language for user
- set privileges for user

Qualifiers:

/CREATE	create entry in rvs® user table
/DELETE	delete entry in rvs® user table
/UPDATE	update entry in rvs® user table

If no qualifier has been specified, the entry is either created or updated depending on whether it already exists or not.

Parameters:

UID	(optional) user ID which identifies the entry to be modified; a value different from the user ID of the current user may be specified only by a privileged user; default: current user
NAME	(optional) new name of user
LANGUAGE	(optional) new language setting (e.g. **E**=English, **D**=Deutsch)
PRIV	(optional) new privilege for user; (**U**=user, **O**=operator, **A**=administrator)

Examples:

```
*---------- create new user (default=own UID)
*
USER /C
*
*---------- create new user
*
USER /C UID=newuser LANGUAGE=d NAME=x PRIV=O
USER /C UID=extrauser
*
*---------- update existing user
*
USER /U UID=newuser LANGUAGE=e NAME=y PRIV=U
*
*---------- delete user
*
USER /D UID=newuser
```

4.2.10 Command `ACTIVATE`

Function:

- Activate a partner station:
- rvs® will start a sender process, which establishes the connection to the partner station. All queued files will be transmitted. After that, the connection is closed.

Parameters:

SID (required) station ID of partner

Examples:

```
*
*---------- activate station ABC
*          (connect and send or receive queued
data sets)
*
ACTIVATE SID=ABC
```

4.2.11 Command `MODST`

Function:

Modify station table.

Attention: **This command overwrites old database entries**; other existing entries will not be deleted (only **DELST** will delete an entry)

Parameters:

DSN (required) name of data set which contains a station table. the file name can be a single input file or a directory which contains several input files.

Examples:

```
*
*---------- modify station table:
*          (open the file (here: UNIX file
name),
*          read contents and put it into
database):
*
MODST DSN="/home/rvs/init/new_rdstat.dat"
```


© gedas deutschland GmbH / Competence Center EDI/rvs®

4.2.12 Command `DELST`

Function:

Delete entry station table.

Parameters:

SID Station ID of station to be deleted

Examples:

```
*
*---------- delete entry ABC in database:
*
DELST SID=ABC
```

4.2.13 Command `LISTPARM`

Function:

- List the value of a rvs® parameter.
- In batch mode, the message "I: value" appears.
- In C programs, the value is given as a string in the `rvscal()` output parameter **S_MSG**.

Parameters:

parameter name of rvs® parameter

Examples:

```
*
*---------- show the value of the parameter SLEEP
*
LISTPARM SLEEP
```

4.2.14 Command SETPARM

Function:

Set the value of a rvs® parameter:

Parameters:

parameter name of rvs® parameter

Examples:

```
*
*---------- set the value of the parameter SLEEP
*
SETPARM SLEEP=2
```


© gedas deutschland GmbH / Competence Center EDI/rvs®

5 How to Work with rvs® C-CAL Interface

The following sections describes how to use the C-language functions that may be linked into an application program to execute rvs® utility commands. Please check `rvscal.h` that is contained in the rvs® distribution for last changes of structures and function prototypes.

Note: For all tasks (sending and receiving data files as well as administration of rvs®) you have to do with rvs® you can either

- call the general function `rvscal()` using string commands or
- use several dedicated functions

The string commands of `rvscal()` are identical to those useable for `rvsbat`. The function `rvscal()` is described in chapter "How to work with rvs® Batch Interface and rvs® C-CAL Interface".

On the other hand the dedicated functions are on a lower abstraction level thus providing more control and flexibility. They are described in the following chapters.

5.1 Sending and Receiving with C-CAL Interface

This chapter describes the functions which are required to manage send entries using the C-CAL Interface. For all this functions a type definition and the corresponding prototypes are.

5.1.1 Type Definitions

```
typedef struct {
  SINT i_error;
  SINT i_type;
  char s_uid [RVSCAL_L_USID];
  char s_jobid [RVSCAL_L_JOBID];
  char sid_neighb [RVSCAL_L_STATID];
  char sid_dest [RVSCAL_L_STATID]; /* destination
SID */
  char sid_sender [RVSCAL_L_STATID];
  char s_vdsn [RVSCAL_L_VDSN];
  char dt_created [RVSCAL_L_DT]; /* job creation
date and time */
  char dt_avail [RVSCAL_L_DT];
  char dt_sched [RVSCAL_L_DT];
  char dt_begin [RVSCAL_L_DT];
  char dt_end [RVSCAL_L_DT];
  char dt_done [RVSCAL_L_DT];
  char dt_received [RVSCAL_L_DT];
  long int cnt_sendatt; /* number of send
attempts */
  long int cnt_record;
  long int cnt_byte; /*number of already sent
bytes*/
  long int cnt_maxrecl;
  long int cnt_apsize;
  long int cnt_lenvm;
  long int cn_se;
  long int cn_sk;
  long int cn_ie;
  long int cn_iz;
  long int cn_re;
  char status_et [RVSCAL_L_KS];
  char status_se [RVSCAL_L_KS];
  char status_sk [RVSCAL_L_KS] ;/* state of send
cmd (-|a|i|f|p|e|d|s) */
  char status_ie [RVSCAL_L_KS];
  char status_iz [RVSCAL_L_KS];
  char dsn_local [RVSCAL_L_DSN];
  char s_recfm [RVSCAL_L_C1];
  char s_ftype [RVSCAL_L_C1];
  char s_code [RVSCAL_L_C1];
  char s_codein [RVSCAL_L_C1];
  char s_codeout [RVSCAL_L_C1];
  char s_disp [RVSCAL_L_C1];
  char s_label [RVSCAL_L_SEUSRLABEL]; /* user
defined label */
  SINT flg_tstamp;
} INFO_SK ;
```


© gedas deutschland GmbH / Competence Center EDI/rvs®

```
/*SetSendEntry Commands: */
#define SET_HOLD        1
#define SET_RELEASE     2
#define SET_DELETE      3


#define TRANSMISSION_SEND 1
#define TRANSMISSION_RECV 2


#define CN_START          0
```

5.1.2 Get next send entry from Database

Prototype `rvsGetNextSend`:

```
PROCDEF int PROCKEYW rvsGetNextSend(long
prev_send_cn, INFO_SK *info);
```

Description of Parameters

FUNCTIONVALUE (int)

=**RVSCAL_OK**, if we found a next send entry

=**RVSCAL_END_FETCH**, if there are no send entries greater than **PREV_SEND_CN**

=**RVSCAL_INTERNAL_ERROR**, if internal database error is occured

prev_send_cn (int, input)

Command no. of previous send entry

info (struct **INFO_SK ***, output)

Struct with informations about next send entry with command number greater than **prev_send_cn**

REMARKS `rvsGetNextSend` looks for the next send entry with a value greater than **prev_send_cn**. When the function is called the first time **prev_send_cn** must be **CN_START. CN_START** caused `rvsGetNextSend` to read all entries from the rvs® database and save it to an internal list. Every call of `rvsGetNextSend` with **prev_send_cn**=**CN_START** will refresh that list.

5.1.3 Get a send entry from Database

Prototype `rvsGetSendEntry`:

```
PROCDEF int PROCKEYW rvsGetSendEntry(const long
send_cn, INFO_SK *info);
```

Description of Parameters

FUNCTIONVALUE (int)

=**RVSCAL_OK**, if we found the informations about this send command

=**RVSCAL_END_FETCH**, if there is no send entry with given command number

=**RVSCAL_INTERNAL_ERROR**, if internal database error occured

send_cn (int, input)

Command number of a send entry

info (struct **INFO_SK ***, output)

Struct with informations about send entry

5.1.4 Set debug mode

Prototype `rvsSetDebugMode`:

```
PROCDEF int PROCKEYW rvsSetDebugMode(int mode);
```

Description of Parameters

FUNCTIONVALUE (int)

=**RVSCAL_OK**, if debug mode is set

=**RVSCAL_INTERNAL_ERROR**, if internal database error is occured

MODE (int, input)

mode type

© gedas deutschland GmbH / Competence Center EDI/rvs®

5.1.5 Change status of SE

Prototype `rvsSetSendEntry`:

```
PROCDEF int PROCKEYW rvsSetSendEntry(long int
cn_se, int SetCmd, char *szSID, char *s_msg);
```

Description of Parameters

FUNCTIONVALUE	(int)
	=**RVSCAL_OK**, if no error occured
	=**RVSCAL_INTERNAL_ERROR**, if internal database error occured
cn_se	(long int, input)
	command no. of SE (returned by `CreateSendEntry`)
SetCmd	(int, input)
	Command type
	SET_HOLD : Hold this send entry
	SET_RELEASE : Release this send entry
	SET_DELETE : Delete this send entry
szSID	(char *, input)
	Station ID of receiver
s_msg	(char *, output)
	Error message text in case of error

5.1.6 Get next information entry

Prototype `rvsGetNextIE`:

```
PROCDEF int PROCKEYW rvsGetNextIE(long int
prev_ie_cn, INFO_SK *p_info);
```

Description of Parameters

FUNCTIONVALUE	(int)
	=**RVSCAL_OK**, if no error occured.
	=**RVSCAL_INTERNAL_ERROR**, if internal database error is occured
prev_ie_cn	(long int, input)
	command no. of previous info entry
p_info	(struct **INFO_SK** *, output)
	data struct containing info about entry

© gedas deutschland GmbH / Competence Center EDI/rvs®

5.1.7 Send a file

Prototype `rvsCreateSendEntry`:

```
PROCDEF int PROCKEYW rvsCreateSendEntry(
    char *dsn,
    char *disp,
    char *format,
    char *codein,
    char *inittime,
    char *serial,
    char *label,
    char *tstamp,
    char *sid,
    char *dsnnew,
char *codeout,
    char *s_msg);
```

Description of Parameters

FUNCTIONVALUE (int)

=**RVSCAL_OK**, if it was successfull

=**RVSCAL_INTERNAL_ERROR**, if error occured

dsn (char *, input)

filename of local file (details: see rvs® User Manual)

disp (char *, input)

disposition (**K**=keep, **D**=delete after transmission)

format (char *, input)

file format (**T**/**F**/**V**/**U**)

codein (char *, input)

code of input file (**A**=ASCII, **E**=EBCDIC)

inittime (char *, input)

time of earliest send attempt (**YY**/**MM**/**DD** **HH**:**MM**:**SS**)

serial (char *, input)

serialization flag (**Y** or **N**)

label (char *, input)

label of file (for serialization)

tstamp	(char *, input)
	timestamp
sid	(char *, input)
	station ID of receiver
dsnnew	(char *, input)
	virtual file name (**OFTP**)
codeout	(char *, input)
	output code (**A**=ASCII, **E**=EBCDIC)
s_msg	(char *, output)
	error message in case of error

5.1.8 Create a Send Entry

Prototype `rvsCreateSendEntryCmd`:

```
PROCDEF int PROCKEYW rvsCreateSendEntryCmd(
    char *dsn,
char *disp,
    char *format,
    char *codein,
    char *inittime,
    char *serial,
    char *label,
    char *tstamp,
    char *sid,
    char *dsnnew,
    char *codeout,
    char *s_msg);
```

Description of Parameters

FUNCTIONVALUE	(int)
	=command number of send entry, if it was successfull
	=**RVSCAL_ERROR_CREATESEND**, if error occured
PARAMETERS	see chapter "
	Send a file"

© gedas deutschland GmbH / Competence Center EDI/rvs®

5.2 Administration with C-CAL Interface

This chapter describes the functions to manage

- Station Table Entries
- rvs® Parameters
- rvs® Operator Commands
- Resident Receive Entries
- Entries for Jobstart after Send Attempt
- User Entries
- Database Functions

For all this functions a type definition is given as well as the corresponding prototypes and return codes.

5.2.1 Functions to manage Station Table Entries

This chapter describes the functions which are required to manage the Station Table entries.

5.2.1.1 Type Definitions

```
typedef struct {
  char            netid[RVSCAL_L_NETID];
  char            statname[RVSCAL_L_STATNAME];
  char            phone[RVSCAL_L_PHONE];
} INFO_ST;


typedef struct {
  char            ftp[RVSCAL_L_C1];
  char            protocol[RVSCAL_L_C1];
  char            autodial[RVSCAL_L_C1];
  SINT            pr_nk;
  SINT            flg_suspnd;
} INFO_NK;


typedef struct {
  char            sidneighb[RVSCAL_L_STATID];
  SINT            pr_rt;
} INFO_RT;
```

```
typedef struct {
  char            odetteid[RVSCAL_L_ODETTEID];
  char            pswfrom[RVSCAL_L_OPSW];
  char            pswto[RVSCAL_L_OPSW];
  long            i_sendblocks;
  long            i_recvblocks;
  long            i_ocreval;
  long            i_oexbuf;
  char            codein[RVSCAL_L_C1];
  char            codeout[RVSCAL_L_C1];
  char            userfield[RVSCAL_L_OFTP_USERD];
  char            eerp_in[RVSCAL_L_OFTP_EERP];
  char            eerp_out[RVSCAL_L_OFTP_EERP];
  char            vdsnchar[RVSCAL_L_OFTP_EERP];
  char            retry[RVSCAL_L_DT];
} INFO_OP;


typedef struct {
  char            netid_lu[RVSCAL_L_LU62_NETID];
  char            luname[RVSCAL_L_LU62_LUNAME];
  char            tpname[RVSCAL_L_LU62_TPNAME];
  char            userid[RVSCAL_L_LU62_UID];
  char            password[RVSCAL_L_LU62_PASSW];
  char            profile[RVSCAL_L_LU62_PROF];
  char            mode[RVSCAL_L_LU62_MODE];
  SINT            i_security;
  SINT            flg_sync;
  SINT            flg_conv;
} INFO_LU;


typedef struct {
  char            alias[RVSCAL_L_X25_ALIAS];
  char            recv_alias[RVSCAL_L_X25_ALIAS];
  long            cntn;
  char            xaddr[RVSCAL_L_X25_ADDR];
  char            subaddr[RVSCAL_L_X25_SUBADDR];
  char            timeout[RVSCAL_L_X25_TIMEOUT];
  char            isdnno[RVSCAL_L_X25_ISDNNO];
  char            link[RVSCAL_L_X25_LINK];
  char            fac[RVSCAL_L_X25_FAC];
  SINT            flgdbit;
  char            cug[RVSCAL_L_X25_CUG];
  SINT            flgreqrev;
  SINT            flgaccrev;
  SINT            flgfastsel;
  char            usrdata[RVSCAL_L_X25_USDTA];
  char            vc[RVSCAL_L_C1];
  long            cntsessions;
} INFO_XP;
```


© gedas deutschland GmbH / Competence Center EDI/rvs®

```
typedef struct {
  char            protocol[RVSCAL_L_C1];
  long            cntn;
  char            inaddr[RVSCAL_L_TCPIP_ADDR];
  SINT            i_port;
  SINT            i_max_in;
  SINT            i_max_out;
  char            security[RVSCAL_L_TCPIP_SEC];
} INFO_TC;


typedef struct {
  char            lx_name[RVSCAL_L_LX_NAME];
  long            lx_len;
  char            lx_val[RVSCAL_L_LX_VALUE];
} INFO_LX;


typedef struct {
  char            sid[RVSCAL_L_STATID];
  int             flg_st;
  int             flg_nk;
  int             flg_rt;
  int             flg_op;
  int             flg_lu;
  int             flg_xp;
  int             flg_tc;
  int             flg_lx;
  INFO_ST         st;
  INFO_NK         nk;
  INFO_RT         rt;
  INFO_OP         op;
  INFO_LU         lu;
  INFO_XP         xp;
  INFO_TC         tc;
  INFO_LX         lx;
} INFO_STATION;
```

5.2.1.2 Get next station entry from Database

Prototype `rvsGetNextStation`:

```
PROCDEF int PROCKEYW rvsGetNextStation(char
*SIDpre, char * SID);
```

Description of Parameters

SIDPRE (char *, input)

SID of previous station

SID (char *, output)

SID of next station found in **ST** Table

5.2.1.3 Update station entry from Database

Prototype `rvsUpdateStation`:

```
PROCDEF int PROCKEYW rvsUpdateStation(
INFO_STATION *info);
```

Description of Parameters

info (**INFO_STATION** *, input)

Struct with informations about station entry

5.2.1.4 Get station entries from Database

Prototype `rvsGetStation`:

```
PROCDEF int PROCKEYW rvsGetStation( char
*psz_SID, INFO_STATION *info);
```

Description of Parameters

psz_SID (char *, input)

SID of station

info (**INFO_STATION** *, output)

Struct with informations about station entry

5.2.1.5 Delete station entry from Database

Prototype `rvsDeleteStation`:

```
PROCDEF int PROCKEYW rvsDeleteStation( char
*psz_SID);
```

Description of Parameters

psz_SID (char *, input)

SID of station

© gedas deutschland GmbH / Competence Center EDI/rvs®

5.2.1.6 Free all suspended Commands

Prototype `rvsFreeSuspendedCommands`:

```
PROCDEF int PROCKEYW rvsFreeSuspendedCommands(
void);
```

Description of Parameters

there is no parameter

5.2.1.7 Return Codes

FUNCTIONVALUE (int)

=**RVSCAL_OK**, if function succeeds.

=**RVSCAL_END_FETCH**, if there are no stations with this **SID** or no entries greater than **SIDPRE**

=**RVSCAL_DBERROR**, if database could not be opened

=**RVSCAL_NEITHER_X_NOR_ISDN**, if neither X25 address nor ISDN address is specified

=**RVSCAL_NEIGHBOR_STATION**, if there are routing links to this station in case of delete

=**RVSCAL_PARAMETER_CHECK**, if at least one parameter is incorrect

=**RVSCAL_INTERNAL_ERROR**, if internal database error is occured

5.2.2 Functions to manage rvs® Parameters

This chapter describes the functions which are required to manage rvs® parameters.

5.2.2.1 Type Definition

```
    typedef struct{
char    s_parm[RVSCAL_L_PARM_NAME];
    SINT    i_type ;
    long    len ;
    char    s_val[RVSCAL_L_PARM_VAL] ;
  } PARM_STRUCT ;
```

5.2.2.2 Get parameter value from Database

Prototype `rvsGetParm`:

```
PROCDEF int PROCKEYW rvsGetParm( char *parm,
PARM_STRUCT *stparm);
```

Description of Parameters

parm (char *, input)

parameter name

stparm (**PARM_STRUCT** *, output)

Struct with informations about parameter entry

5.2.2.3 Get next parameter from Database

Prototype `rvsGetNextParm`:

```
PROCDEF int PROCKEYW  rvsGetNextParm( char *parm,
PARM_STRUCT *stparm);
```

Description of Parameters

parm (char *, input)

previous parameter

stparm (**PARM_STRUCT** *, output)

Struct with informations about the next parameter entry

© gedas deutschland GmbH / Competence Center EDI/rvs®

5.2.2.4 Writes parameter value into Database

Prototype `rvsWriteParm`:

```
PROCDEF int PROCKEYW rvsWriteParm( char *parm,
PARM_STRUCT *stparm);
```

Description of Parameters

parm (char *, input)

parameter name

stparm (**PARM_STRUCT** *, input)

Struct with informations about parameter entry

5.2.2.5 Return Codes

FUNCTIONVALUE (int)

=**RVSCAL_OK**, if function succeeds

=**RVSCAL_PARAMETER_CHECK**, if at least one parameter is incorrect

=**RVSCAL_INVALID_NAME**, if parameter name doesn't exists

=**RVSCAL_INTERNAL_ERROR**, if internal database error is occured

5.2.3 Functions to manage rvs® Operator Commands

This chapter describes the functions which are required to manage rvs® Operator commands.

5.2.3.1 Store operator command into Database

Prototype `rvsStoreOK`:

```
PROCDEF int PROCKEYW  rvsStoreOK( char *command);
```

Description of Parameters

command (char *, input)

command string

5.2.3.2 Wake the Monitor

Prototype `rvsWakeMonitor`:

```
PROCDEF int PROCKEYW  rvsWakeMonitor( void);
```

Description of Parameters

there are no parameter

5.2.3.3 Return Codes

FUNCTIONVALUE (int)

=**RVSCAL_OK**, if function succeeds

=**RVSCAL_PARAMETER_CHECK**, if at least one parameter is incorrect

=**RVSCAL_INVALID_NAME**, if parameter name doesn't exists

=**RVSCAL_RC_WAKE_FAILED**, if wake command failes

=**RVSCAL_INTERNAL_ERROR**, if internal database error is occured

© gedas deutschland GmbH / Competence Center EDI/rvs®

5.2.4 Functions to manage Resident Receive Entries

This chapter describes the functions which are required to manage Resident Receive Entries.

5.2.4.1 Type Definition and Macros

```
    typedef struct{
    char    uid_local[RVSCAL_L_USID];
    char    vdsn[RVSCAL_L_VDSN];
    char    uid_sender[RVSCAL_L_USID];
    char    sid_sender[RVSCAL_L_STATID];
    char    dsn_local[RVSCAL_L_DSN];
    char    s_replace[RVSCAL_L_C1];
    char    s_disp[RVSCAL_L_C1];
    SINT    flg_stamp;
    char    s_printdef[RVSCAL_L_C1];
    char    dsn_batchjob[RVSCAL_L_DSN];
    char    uid_creator[RVSCAL_L_USID];
    char    accnt_rcv[RVSCAL_L_ACCT];
    char    comment[RVSCAL_L_RECMNT];
    char    dt_lastused[RVSCAL_L_DT];
  } INFO_RE ;


#define RE_UPDATE        1
#define RE_DELETE        2
#define RE_CREATE        3
```

5.2.4.2 Get next command number of Resident Receive Entry from Database

Prototype `rvsGetNextRE`:

```
PROCDEF int PROCKEYW  rvsGetNextRE( const long
cn_pre, long *lpcn_re);
```

Description of Parameters

cn_pre (const long, input)

command number of previous resident receive entry

lpcn_re (long *, output)

command number of next resident receive entry found in **RE** table

5.2.4.3 Get Resident Receive Entry from Database

Prototype `rvsGetRE`:

```
PROCDEF int PROCKEYW  rvsGetRE( const long cn_re,
INFO_RE *reinfo);
```

Description of Parameters

cn_re	(const long, input)
	command number of resident receive entry
reinfo	(**INFO_RE** *, output)
	struct with informations about resident receive entry

5.2.4.4 Configure Resident Receive Entries

Prototype `rvsResidentResceiveEntry`:

```
   PROCDEF int PROCKEYW  rvsResidentReceiveEntry(
const int icmd, INFO_RE *reinfo);
```

Description of Parameters

icmd	(const int, input)
	command to specify what should be done
	RE_UPDATE - updates a resident receive entry **RE_DELETE** - deletes a resident receive entry **RE_CREATE** - creates a resident receive entry
reinfo	(INFO_RE *, input)
	struct with informations about resident receive entry

© gedas deutschland GmbH / Competence Center EDI/rvs®

5.2.4.5 Return Codes

FUNCTIONVALUE (int)

=**RVSCAL_OK**, if function succeeds

=**RVSCAL_END_FETCH**, if there is no matching resident receive entry

=**RVSCAL_PARAMETER_CHECK**, if at least one parameter is incorrect

=**RVSCAL_INVALID_DSN**, if invalid **DSNNEW** was spezified

=**RVSCAL_INVALID_JOB**, if invalid JOB was spezified

=**RVSCAL_INVALID_NAME**, if parameter name doesn't exists

=**RVSCAL_INVALID_SID**, if **SID_SENDER** isn't known

=**RVSCAL_NOT_PRIVILEGED**, if user is not privileged in to configure resident receive entries

=**RVSCAL_DBERROR**, if database could not be opened or closed

=**RVSCAL_RE_NOT_FOUND**, if resident receive entry could not be found

=**RVSCAL_DUPLICATE_RE**, if duplicate resident receive entry has found in case of **icmd**=**RE_CREATE**

=**RVSCAL_INTERNAL_ERROR**, if internal database error is occured

5.2.5 Functions to manage Entries for Jobstart after Send Attempt

This chapter describes the functions which are required to manage Entries for Jobstart after Send Attempt.

5.2.5.1 Type Definition and Macros

```
    typedef struct{
    char    vdsn[RVSCAL_L_VDSN];
    char    uid_sender[RVSCAL_L_USID];
    char    sid_receiver[RVSCAL_L_STATID];
    long    cnt_sendatt;
    char    dsn_batchjob[RVSCAL_L_DSN];
    char    uid_creator[RVSCAL_L_USID];
    char    comment[RVSCAL_L_RECMNT];
    char    dt_lastused[RVSCAL_L_DT];
  } INFO_JS ;
#define JS_UPDATE        1
#define JS_DELETE        2
#define JS_CREATE        3
```

5.2.5.2 Get next command number of Job Start Entry from Database

Prototype `rvsGetNextJS`:

```
PROCDEF int PROCKEYW  rvsGetNextJS( const long
cn_pre, long *lpcn_js);
```

Description of Parameters

cn_pre	(const long, input)
	command number of previous entry
lpcn_js	(long *, output)
	command number of next entry for jobstart after a send request found in **JS** table

© gedas deutschland GmbH / Competence Center EDI/rvs®

5.2.5.3 Get Jobstart Entry from Database

Prototype `rvsGetJS`:

```
PROCDEF int PROCKEYW  rvsGetJS( const long cn_js,
INFO_JS *jsinfo);
```

Description of Parameters

cn_js	(const long, input)
	command number of jobstart entry
jsinfo	(**INFO_JS** *, output)
	struct with informations about jobstart entry

5.2.5.4 Configure Entries for Jobstart after Send Attempt

Prototype `rvsJobStartEntry`:

```
PROCDEF int PROCKEYW  rvsJobStartEntry( const int
icmd, INFO_JS *jsinfo);
```

Description of Parameters

icmd	(const int, input)
	command to specify what should be done **JS_UPDATE** - updates a jobstart entry **JS_DELETE** - deletes a jobstart entry **JS_CREATE** - creates a jobstart entry
jsinfo	(**INFO_JS** *, input)
	struct with informations about job start entry

5.2.5.5 Return Codes

FUNCTIONVALUE (int)

=**RVSCAL_OK**, if function succeeds

=**RVSCAL_END_FETCH**, if there is no matching jobstart entry

=**RVSCAL_PARAMETER_CHECK**, if at least one parameter is incorrect

=**RVSCAL_INVALID_DSN**, if invalid **DSNNEW** was spezified

=**RVSCAL_INVALID_JOB**, if invalid **JOB** was spezified

=**RVSCAL_INVALID_NAME**, if parameter name doesn't exists

=**RVSCAL_INVALID_SID**, if **SID_RECEIVER** isn't known

=**RVSCAL_NOT_PRIVILEGED**, if user is not privileged in to configure entries for Jobstart after Send Attempt

=**RVSCAL_DBERROR**, if database could not be opened or closed

=**RVSCAL_JS_NOT_FOUND**, if job start entry could not be found

=**RVSCAL_DUPLICATE_JS**, if duplicate jobstart entry was found in case of **icmd** = **JS_CREATE**

=**RVSCAL_INTERNAL_ERROR**, if internal database error is occured

© gedas deutschland GmbH / Competence Center EDI/rvs®

5.2.6 Functions to manage User Entries

This chapter describes the functions which are required to manage User Entries.

5.2.6.1 Type Definition and Macros

```
typedef struct{
    char    uid[RVSCAL_L_USID];
    char    s_priv[RVSCAL_L_C1];
    char    s_prof[RVSCAL_L_C1];
    char    s_lang[RVSCAL_L_LANG];
    char    s_name[RVSCAL_L_USERNAME];
  } INFO_USER ;


#define USER_UPDATE        1
#define USER_DELETE        2
#define USER_CREATE        3
```

5.2.6.2 Get next User from Database

Prototype `rvsGetNextUser`:

```
PROCDEF int PROCKEYW  rvsGetNextUser( char
*userpre, char *user);
```

Description of Parameters

userpre	(char *, input)
	previous user name
user	(char *, output)
	next user name found in **BT** table

5.2.6.3 Get User Entry from Database

Prototype `rvsGetUser`:

```
PROCDEF int PROCKEYW  rvsGetUser( char *user,
INFO_USER *usinfo);
```

Description of Parameters

user	(char *, input)
	user name
usinfo	(**INFO_USER** *, output)
	struct with informations about user entry

5.2.6.4 Configure User Entries

Prototype `rvsUser`:

```
PROCDEF int PROCKEYW  rvsUser( int icmd,
INFO_USER *usinfo);
```

Description of Parameters

icmd (const int, input)

command to specify what should be done **USER_UPDATE** - updates a user **USER_DELETE** - deletes a user **USER_CREATE** - creates a user

usinfo (**INFO_USER** *, output)

struct with informations about user entry

5.2.6.5 Return Codes

FUNCTIONVALUE (int)

=**RVSCAL_OK**, if function succeeds

=**RVSCAL_END_FETCH**, if there is no matching user entry

=**RVSCAL_PARAMETER_CHECK**, if at least one parameter is incorrect

=**RVSCAL_INVALID_OWN_PRIV**, if user wants to decrease his own privileges

=**RVSCAL_INVALID_UID**, if parameter **UID** is empty

=**RVSCAL_INVALID_USER**, if user does'nt exist or wants to delete himself

=**RVSCAL_NOT_PRIVILEGED**, if user is not privileged in to configure user entries

=**RVSCAL_DBERROR**, if database could not be opened or closed

=**RVSCAL_USER_NOT_FOUND**, if user entry could not be found

=**RVSCAL_DUPLICATE_USER**, if duplicate user entry has found in case of **icmd** = **USER_CREATE**

=**RVSCAL_INTERNAL_ERROR**, if internal database error is occured

© gedas deutschland GmbH / Competence Center EDI/rvs®

5.2.7 rvs® Database Functions

This chapter describes the functions which are required to manage rvs® Database functions.

5.2.7.1 Type Definition and Macros

```
#define RVSCAL_PIPE_NAME "\\\\.\\pipe\\rvsdb"
#define RVSCAL_OLEVENT_NAME "rvsdb_olevent"
#define RVSCAL_PIPE_TIMEOUT     90000
#define RVSCAL_L_OLEVENT        14
#define RVSCAL_L_PIPENAME       15
#define DEL_DB                  0x01
#define DEL_LOG                 0x02
#define DEL_TMP                 0x04
#define DEL_REMDB               0x08
#define DUMP_RES                0x01
#define DUMP_USER               0x02
#define DUMP_JS                 0x04
#define DUMP_STATION            0x08
#define DUMP_RU_ALL            DUMP_RES |
DUMP_USER | DUMP_JS | DUMP_STATION
```

5.2.7.2 Dump Database

Prototype `rvsDumpDB`:

```
PROCDEF int PROCKEYW  rvsDumpDB(  char
*environment, char *dsn);
```

Description of Parameters

environment (char *, input)

name of the environment data set, If environment is **NULL** or an empty zero terminated string rvs® will look for the default environment

dsn (char *, input)

data set name of dump file

5.2.7.3 Recover Database

Prototype `rvsWriteDB`:

```
PROCDEF int PROCKEYW  rvsWriteDB( char
*environment, char *dsn);
```

Description of Parameters

environment	(char *, input) name of the environment data set, If environment is **NULL** or an empty zero terminated string rvs® will look for the default environment
dsn	(char *, input) data set name of dump file

5.2.7.4 Initialize Database

Prototype `rvsInitDB`:

```
PROCDEF int PROCKEYW  rvsInitDB( char
*environment, char *sid_loc);
```

Description of Parameters

environment	(char *, input) name of the environment data set, If environment is **NULL** or an empty zero terminated string rvs® will look for the default environment
sid_loc	(char *, input) local station ID

© gedas deutschland GmbH / Competence Center EDI/rvs®

5.2.7.5 Delete Database

Prototype `rvsDeleteDB`:

```
PROCDEF int PROCKEYW  rvsDeleteDB( char
*environment, const int delattrib);
```

Description of Parameters

environment (char *, input)

name of the environment data set, If environment is **NULL** or an empty zero terminated string rvs® will look for the default environment

delattrib (const int, input)

attribute indicates which files should be deleted, possible values are

DEL_DB (default) - remove all database files,

DEL_LOG - remove *.logfiles,

DEL_TMP - remove all files out of the temporary directory of rvs®

5.2.7.6 Dump User-, Receive, Js- and Station Entries

Prototype `rvsDumpRU`:

```
PROCDEF int PROCKEYW  rvsDumpRU( char
*environment, char *dsn, const int dumpattrib);
```

Description of Parameters

environment (char *, input)

name of the environment data set, If environment is **NULL** or an empty zero terminated string rvs® will look for the default environment

dsn (char *, input)

data set name of dump file. The data set contents will get the input format useable with the rvs® batch interface.

dumpattrib (const int, input)

attribute indicates which files should be dumped, possible values are

DUMP_USER (default) - dump all user entries,

DUMP_RES - dump all entries for Jobstart after Send request **DUMP_JS** - **DUMP_STATION**

5.2.7.7 Return Codes

FUNCTIONVALUE (int)

=**RVSCAL_OK**, if function succeeds

=**RVSCAL_PARAMETER_CHECK**, if at least one parameter is incorrect

=**RVSCAL_DSN_NOT_EXIST**, if given data set does not exist

=**RVSCAL_INTERNAL_ERROR**, if internal database error is occured

5.2.7.8 Get the version of rvs® Database

Prototype `rvsGetVersion`:

```
PROCDEF int PROCKEYW rvsGetDBVersion( char
*pszDBVersion);
```

Description of Parameters

pszDBVersion (char *, output)

version of rvs® database

5.2.7.9 Return Codes

FUNCTIONVALUE (int)

=**RVSCAL_OK**, if function succeeds

=**RVSCAL_PARAMETER_CHECK**, if at least one parameter is incorrect

=**RVSCAL_ENVIRONMENT_NOT_EXIST**, if environment data set does not exist

=**RVSCAL_ERROR_GETVERSION**, if version couldn't be determined

© gedas deutschland GmbH / Competence Center EDI/rvs®

5.2.8 Other Functions

This chapter describes how to get SID from ODETTE and vice versa as well as how to get the list status of rvs® commands.

5.2.8.1 Get SID from ODETTE ID or vice versa

Prototype `rvsgetsid`:

```
char *rvsgetsid(char *s_odette_id);
```

Description of Parameters

FUNCTIONVALUE (char *)
=**NULL**, if no **SID** found or **DB** error.
=pointer to string containing rvs® **SID**

s_odette_id (char *, input)
pointer to string containing ODETTE ID (max 26 bytes)

Prototype `rvsgetodid`:

```
char *rvsgetodid(char *s_sid);
```

Description of Parameters

FUNCTIONVALUE (char *)
=**NULL**, if no ODETTE ID found or **DB** error
=pointer to max. 26 byte string

s_sid (char *, input)
pointer to string containing rvs® SID

5.2.8.2 List status of rvs® Commands

Prototype `rvslistcmd`:

```
typedef struct {
    char  status;
    short errorcode;
    } RVSCMD;
int rvslistcmd(long  l_cmdid, RVSCMD *p_info);
```

Description of Parameters

FUNCTIONVALUE	(int) =**RVSCAL_OK**, if `rvslistcmd` succeeded
l_cmdid	(long, input) the **CMDID** (command number) of the command which was processed (return value of `rvscal`)
p_info	(struct **RVSCMD** *, output) pointer to a struct which contains information about the command **CMDID**
RVSCMD.status	(char, output) character which contains the status of the command: **a/d/e/h/p/q/s/f/...** (see chapter " Display of Transmissions (send) (mask DDT1)")
RVSCMD.errorcode	(short, output) short integer which contains an error code, if it is different from **0**

© gedas deutschland GmbH / Competence Center EDI/rvs®

Glossary

A

Access Method	The access method describes the way by which two stations are connected.
ASCII	American National Standard Code for Information Interchange

B

Batch Interface (`rvsbat`)	The batch interface of rvs® offers user functionality for automatic background use.

C

Communication Module (`rvscom`)	The communication module of the rvs® system connects to another station and sends or receives files.

D

Dialogue Interface (`rvsdia`)	The dialogue interface of rvs® provides interactiv user functionality.

E

EBCDIC	Extended Binary Coded Decimal Interchange Code
EDI	**E**lectronic **D**ata **I**nterchange
EDIFACT	**E**lectronic **D**ata **I**nterchange for **A**dministration **C**ommerce and **T**ransport
EERP	End-to-End-Response; ODETTE expression
ET	(`EmpfängerTabelle'; table of recipients) internal rvs® table describing one recipient
ETSI	European Telecommunications Standards Institute

F

G

H

HPFS	High Performance File System (OS/2)

I

IE	(`InformationsEingang'; information reception) internal rvs® command controlling delivery and routing of received files
IZ	(`InformationsZustellung'; information delivery) internal rvs® command delivering received files to one local recipient

J

K

L

M

MasterTransmitter (`rvsxmt`)	The MasterTransmitter of the rvs® system coordinates send and receive processes to ensure the optimal use of the net capacity.
Monitor (`rvsmon`)	The monitor is the main task of a rvs® system. It controls all other processes and initiates automatic follow up jobs if necessary.

N

O

ODETTE	Organization for Data Exchange by Tele Transmission in Europe

© gedas deutschland GmbH / Competence Center EDI/rvs®

OFTP	ODETTE File Transfer Protocol The ODETTE File Transfer Protocol is the definition of a file transfer protocol by the ODETTE Group IV for OSI Layers 4 to 7. International Protocol used in many business fields (Industry, Commerce, Finance, ..).
Operator Console (`rvscns`)	The operator console provides the administrator with rvs® functions to control the rvs® system.
OSI	Open System Interconnection

P

PDF	Portable Document Format
Protocol	To connect two different computers they have to follow the same protocol. This protocol defines actions and reactions as well as the ”language” spoken.

Q

R

RE	(`Residenter Empfangseintrag'; resident receive entry) rvs® table describing actions to take, when a particular data set is delivered to a local recipient
`rvsmon`	See Monitor

S

SE	(`SendeEintrag'; send entry) internal rvs® command controlling sending of files
Send Entry	Order to rvs® which file has to be sent to which station. This entry is saved in the database.
SID	Station identification, rvs® internal name of an OFTP partner station

SK	(`SendeKommando'; send command) internal rvs® command controlling transfer of one file to one neighboring node
Station	A station is a node that can be addressed within a rvs® network. Each station is identified by a unique station ID (SID).

T

Transfer Component	Control program and line driver for a special access method

U

V

VDA	Verband der Deutschen Automobilhersteller Adress: Verband der Automobilindustrie e.V. (VDA) Abt. Logistik Postfach 17 05 63 60079 Frankfurt Tel.: 069-7570-0
VDSN	Virtual Data Set Name name under which file is known during transfer and for delivery

W

X

Y

Z

© gedas deutschland GmbH / Competence Center EDI/rvs®

Index

Basic Functional Characteristics 8

Confirm

- delete resident receive entry ... 71
- deletion of job start after send attempt entry 85

Confirm deletion of

- send request 58
- user entry 92

Create

- job start after send attempt entry 80
- JS 111
- resident receive entry 64
- Resident Receive Entry 107
- user entry 90

data conversion 9

Delete

- JS 113
- RE 109
- sent request 57

Display

- files sent 99
- job start after send attempt entry 82
- received files 100
- resident receive entry 68
- Resident Receive Entry 106
- user entry 91

Display / Delete

- Transmissions 41

Display modes 31

Display of

- Transmissions 44, 47, 51, 53

EDI 6, 168

Exit rvsNT 34

Function `rvscal`

- prototype 121

functional characteristics 8

Handling of Commands ...117

Information about

- rvs 16, 44, 51, 95, 168
- rvs® 95

Job start

- after send attempt entries 77

key characteristics 8

List of

- data sets for transmission 58
- job start after send attempt entries 78
- user entries 88

Lists of

- stations 39

Macro definitions 118

mask

- CDRE 71
- CDSE1 58
- CJSS 80
- CRRE 64
- CUSR 90
- DDT0 41
- DDT1 44
- DDTR 51
- DELSR 57
- DJSS 85
- DSNL 58
- DSPR 53
- DSPT 47
- DUSR 92
- INFO 95
- INI 27
- JS 77
- JSSL 78
- RE 60
- RE1 68
- REL 62
- REU 73
- SIDL 39
- SJSS 82
- SND 35
- SNDDL 40
- SUSR 91
- UJSS 86
- USRL 88
- UUSR 93

MasterTransmitter 169

Modify

- JS 112
- RE 109

Monitor commands 170

New user 114

physical network..... 6, 7, 171
Platforms 7
portable version..... 6
Prototype
rvsCreateSendEntry 144
rvsCreateSendEntryCmd 145
rvsDeleteDB 164
rvsDeleteStation 149
rvsDumpDB 162
rvsDumpRU 164
rvsFreeSuspendedCommands 150
rvsGetJS..... 158
rvsGetNextIE 143
rvsGetNextJS 157
rvsGetNextParm..... 151
rvsGetNextRE 154
rvsGetNextSend 140
rvsGetNextStation 148
rvsGetNextUser..... 160
rvsgetodid..... 166
rvsGetParm 151
rvsGetRE..... 155
rvsGetSendEntry..... 141
rvsgetsid 166
rvsGetStation..... 149
rvsGetUser 160
rvsGetVersion..... 165
rvsInitDB..... 163
rvsJobStartEntry..... 158
rvslistcmd 167
rvsResidentResceiveEntry 155
rvsSetDebugMode..... 141
rvsSetSendEntry..... 142
rvsStoreOK..... 153
rvsUpdateStation..... 149
rvsUser 161
rvsWakeMonitor 153
rvsWriteDB 163
rvsWriteParm..... 152
Receive17, 31, 33, 60, 62, 64, 68, 71, 73, 154
Resident Receive Entries17, 60, 62, 154
rvs® database 7, 8
Save..... 115
Send12, 20, 31, 32, 33, 40, 41, 57, 58, 77, 78, 80, 82, 85, 86, 157, 158, 159, 164, 170
Send data set
distribution list..... 40
Send order
modify 101
release 101
remove..... 102
restart..... 102
temporarily hold 101
SID 171
Sort..... 114
Status display..... 32
Time stamping..... 15
Transmission of
data set..... 35
Update
job start after send attempt entry 86
resident receive entry 73
user entry..... 93
Update display 99, 100
User
modify 115
new 114
save 115
user interface 8
User Modification 115
User table
modify 114
What rvs® is..... 6
What rvs® is not..... 7
Window areas 32

© gedas deutschland GmbH / Competence Center EDI/rvs®